The Dissertation Committee for John Austin Cottrell, III
certifies that this is the approved version of the following dissertation:

Isogeometric Analysis and Numerical Modeling of the Fine Scales within the Variational Multiscale Method

Committee:

Thomas J. R. Hughes, Supervisor

Graham F. Carey

Björn Engquist

Omar Ghattas

J. Tinsley Oden

Isogeometric Analysis and Numerical Modeling of the Fine Scales within the Variational Multiscale Method

by

John Austin Cottrell, III, B.S., M.S.

Dissertation

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

Doctor of Philosophy

The University of Texas at Austin

August 2007

For Ayiesha.

Acknowledgments

My many thanks go out to those who have believed in me and who have made me believe in myself. In particular:

Prof. Thomas J.R. Hughes, who has consistently made me feel as though my contributions were important to the group, the institute, and the field – thank you for your inspiring example, generous support, and patient understanding. Ayiesha, my brilliant, beautiful, wonderful wife – with you nothing is impossible. Johnny and Elizabeth, my parents, and Sarah, my sister – you all would be proud of me no matter what I do; I hope to always be worthy of that support. Yuri, Victor, Alessandro, Giancarlo, Christian, and Thomas, my friends and colleagues – thank you for the friendship first, and for all of your help second.

JOHN AUSTIN COTTRELL, III

The University of Texas at Austin
August 2007

Isogeometric Analysis and Numerical Modeling of the Fine Scales within the Variational Multiscale Method

Publication No. _____

John Austin Cottrell, III, Ph.D.
The University of Texas at Austin, 2007

Supervisor: Thomas J. R. Hughes

This work discusses isogeometric analysis as a promising alternative to standard finite element analysis. Isogeometric analysis has emerged from the idea that the act of modeling a geometry exactly at the coarsest levels of discretization greatly simplifies the refinement process by obviating the need for a link to an external representation of that geometry. The NURBS based implementation of the method is described in detail with particular emphasis given to the numerous refinement possibilities, including the use of functions of higher-continuity and a new technique for local refinement. Examples are shown that highlight each of the major features of the technology: geometric flexibility, functions of high continuity, and local refinement.

New numerical approaches are introduced for modeling the fine scales within the variational multiscale method. First, a general framework is presented for seeking solutions to differential equations in a way that approximates optimality in certain norms. More importantly, it makes possible

for the first time the approximation of the fine-scale Green's functions arising in the formulation, leading to a better understanding of machinery of the variational multiscale method and opening new avenues for research in the field. Second, a simplified version of the approach, dubbed the "parameter-free variational multiscale method," is proposed that constitutes an efficient stabilized method, grounded in the variational multiscale framework, that is free of the *ad hoc* stabilization parameter selection that has plagued classical stabilized methods. Examples demonstrate the efficacy of the method for both linear and nonlinear equations.

Contents

List of Figures

Chapter 1

Introduction

This dissertation introduces the isogeometric analysis concept and its initial implementation using Non-Uniform Rational B-Splines (NURBS), and chronicles its early development. This approach to finite element analysis utilizes shape functions capable of exactly representing complex real-world geometries at every level of discretization. This dissertation also examines the variational multiscale method and introduces novel numerical techniques for approximating the fine scales that are missing from standard finite element solutions. Specifically, numerical approximations of the fine-scale Green's function are used to obtain numerical expressions for the fine-scale solution field. A new approach emerges that we dub the "parameter-free variational multiscale method."

1.1 Background and motivations

1.1.1 Isogeometric analysis

The concept of isogeometric analysis introduced by Hughes, Cottrell and Bazilevs [38] and expanded by Cottrell *et al.* [18, 19], Bazilevs *et al.* [5, 6], and Zhang *et al.* [59] was first motivated by the tremendous gap between Computer Aided Design (CAD) and Finite Element Analysis (FEA). This gap manifests itself in several ways. The first is in the mesh generation process. The object of interest is encapsulated in some type of CAD model. This model often includes ambiguities, such as gaps and overlaps, and levels of detail (*e.g.*, individual bolts, welds, etc.) that make it inappropriate for analysis. Through some process these ambiguities must be removed and defeaturing must be performed to arrive at an Analysis Suitable Geometry (ASG) that exactly encapsulates the features of interest for the calculation (see Figure 1.1). This ASG must then be replaced with a finite element mesh, usually a piecewise polynomial approximation of the actual geometry. This mesh generation

process is incredibly time consuming and is one of the major bottlenecks in the analysis process.[1]

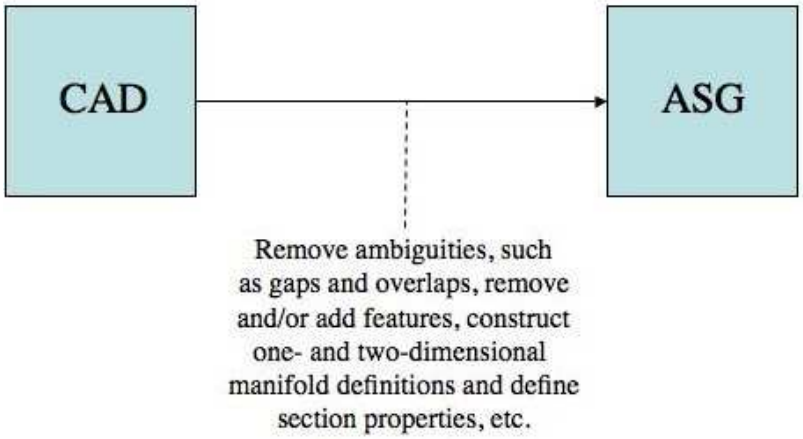


Figure 1.1: The geometry of an object of engineering interest is initially encapsulated in a Computer Aided Design (CAD) package. The CAD description must frequently be changed significantly to create an Analysis Suitable Geometry (ASG).

Another problem arising from the gap between design and analysis is that mesh generation introduces errors into the geometry. There are cases, as in Figure 1.2, where a faceted approximation to smooth geometry can introduce spurious oscillations into the solution. Another example, shown in Figure 1.3, shows that there are times when, if an accurate solution is to be obtained through a series of refinements, the quality of the geometric approximation must be improved simultaneously or else the error can reach a plateau beyond which it will not improve. If such geometric refinement is to take place, a link must be established between the ASG and the refinement package. This link frequently does not exist, or is not practical (see Figure 1.4a). This may be one of the reasons why automatic refinement has had so little impact in industry despite its great success in academia.

Isogeometric analysis is a methodology for addressing these problems. The goal is to have one and only one representation of the geometry which exactly encapsulates the ASG and is more faithful to the underlying CAD technology. While an ASG must still be constructed, using functions and technologies of the sort found in CAD packages may facilitate the development of links between the design and analysis software. More importantly, if the finite element mesh could exactly encapsulate the ASG, refinement to any level could take place completely within the analysis framework. The need for reestablishing the link with an external description of the geometry would be completely obviated as the mesh would *be* the exact geometry, as in Figure 1.4b.

[1] For example, during the ICES seminar on September 29, 2005, David Young of Boeing noted that in a Computational Fluid Dynamics (CFD) study that had taken a total of nine months, more than five of those months had been spent on developing a quality mesh.

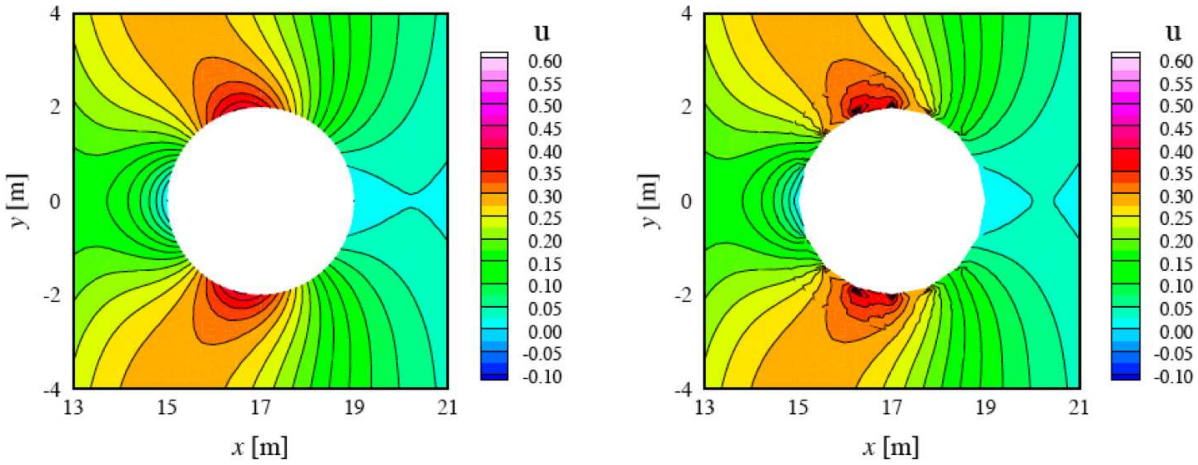


Figure 1.2: The 2D Boussinesq equations. The x-component of velocity solved for using 552 triangles with fifth order polynomials on each triangle. On the left, the cylinder is approximated by elements with curved edges. On the right, the elements are straight sided. The spurious oscillations in the solution on the right are due to the use of straight sided elements (From Eskilsson and Sherwin [23]).

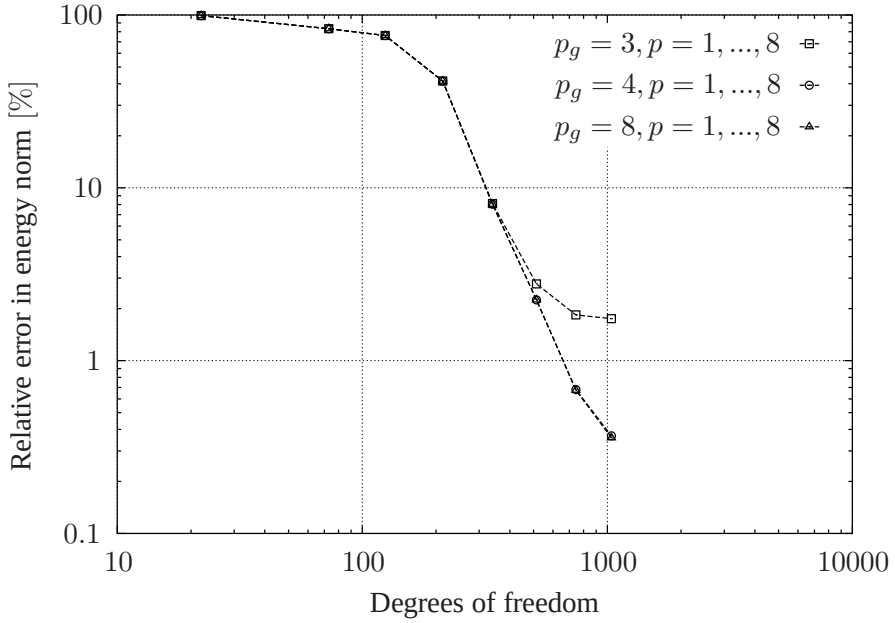


Figure 1.3: Convergence study of the Scordelis-Lo roof problem: p_g represents the polynomial degree of geometry representation, p corresponds to the polynomial degree of the approximation space (from Rank *et al.* [47]).

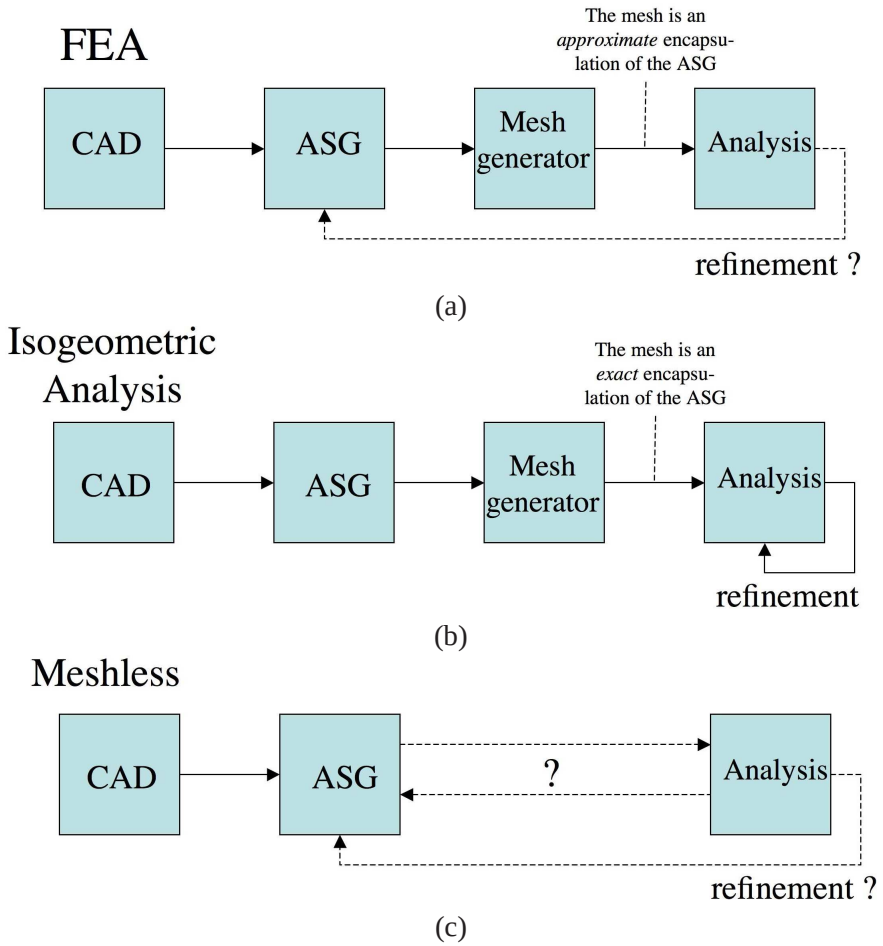


Figure 1.4: The analysis process. a) In finite element analysis, mesh refinement requires interaction with an external description of the geometry if the quality of the geometric approximation is to be improved. The lack of such interaction is an impediment to adaptive mesh refinement procedures. b) In isogeometric analysis, the mesh *is* the exact geometry and so refinement can take place completely within the analysis framework. c) The literature on meshless methods is yet to present a comprehensive view how complex geometries may be represented and how that representation interacts with the process of refining the solution space.

Our current implementation of the isogeometric analysis concept, based on Non-Uniform Rational B-Splines (NURBS), accomplishes this last task in almost all situations. The geometric flexibility of the NURBS basis allows for the exact representation of a much larger class of objects than is possible with standard finite element technology. Most notably, all conic sections can be represented exactly. Little work has been done up to this point on integrating the isogeometric mesh generation process with existing CAD technologies but, as NURBS are a standard in the CAD industry and already found in many packages[2], there is hope for progress in this direction in the future. At this point, generation of the initial mesh can still be a time consuming process but once it has been performed, the isogeometric mesh *encapsulates the exact geometry and may be refined to any level without ever altering this geometry in any way.*

Meshless methods do seem to share certain features with the isogeometric approach. The description of complicated geometries within such methods, however, has been almost entirely ignored in the literature. Notable exceptions are found in the papers of Subbarayan and colleagues [45, 58] and the recent work of Simkins *et al.* [51]. While meshless methods do show great promise in certain areas, a clear view of the proper way in which to define a geometry, as well as how that description affects both refinement of the solution space and, perhaps more importantly, numerical integration of the basis functions, is yet to emerge; see Figure 1.4c.

An important feature of the NURBS based approach to isogeometric analysis that was not one of the initial motivations for the work is the ability to use functions of higher order *and* higher continuity. Chapter 2 will describe the construction of NURBS basis functions that may have up to $p-1$ continuous derivatives across element boundaries, where p is the order of the underlying polynomial. This is seen in Chapter 3 to have a profound effect in structural vibration problems. The NURBS functions of higher continuity offer a much more compact representation of the vibrational spectra of structures than do standard finite element functions, yielding much greater accuracy per degree of freedom, even at the same polynomial order.

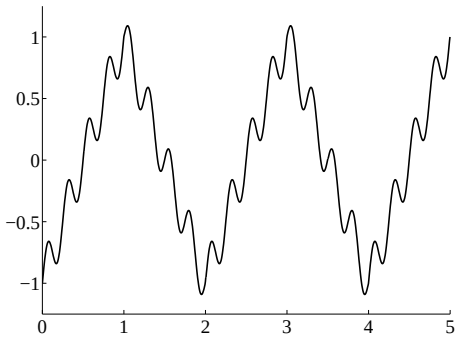
1.1.2 Numerical modeling of the fine scales within the variational multiscale method

The variational multiscale (VMS) method was introduced by Hughes [31] in 1995 as a framework for incorporating the missing fine-scale effects into numerical problems governing coarse-scale behavior. It provides a crucial link to the stabilized methods which predate it, putting them on firm theoretical ground and providing both insight into their success and a direction for subsequent research into their improvement. The approach combines ideas of physical modeling with numerical approximation within a coherent framework.

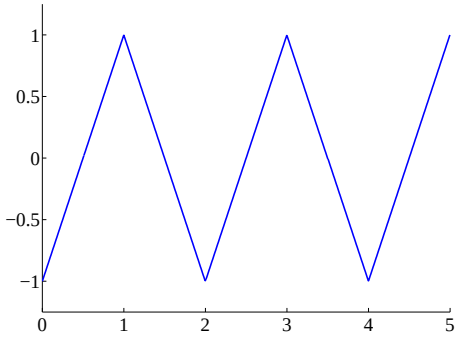
The essence of VMS is simple: consider sum decompositions of the exact (unknown) so-

[2]Note that bivariate NURBS, *i.e.* surfaces, are common in CAD. Analysis requires trivariate NURBS solids.

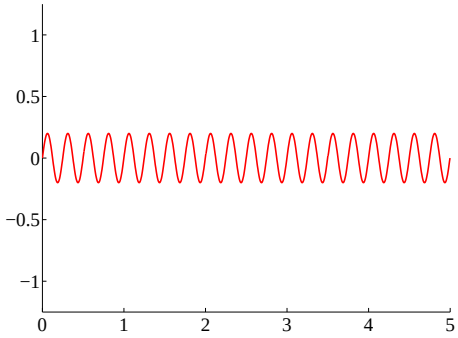
lution, $u = \bar{u} + u'$, as shown schematically in Figure 1.5. We identify the *coarse-scale solution*, $\bar{u}$, with our numerical approximation, and we think of the *fine-scale solution*, u', as the part of the solution that our basis fails to represent.



(a) $u = \bar{u} + u'$, the exact solution



(b) $\bar{u}$, the coarse-scale solution



(c) u', the fine-scale solution

Figure 1.5: A multiscale decomposition of a function u into its coarse-scale component $\bar{u}$, given here by piecewise linear interpolation, and its fine-scale component $u' = u - \bar{u}$.

In its original form, the VMS approach was to seek numerical representations of $\bar{u}$, but to attempt to determine u' *analytically*, eliminating it from the problem for $\bar{u}$. The most common technique for doing so has been to follow the path illuminated by many years of research in stabilized methods and to approximate u' by a scaling parameter, τ, multiplying the residual of the coarse scales. While fairly successful, this method requires a somewhat *ad hoc* selection of τ based on scaling arguments, if not trial and error, and does not clearly point out directions for improvement of the overall method.

The recent paper by Hughes and Sangalli [39] marks a distinct departure from this approach by deriving an analytical expression for the *fine-scale Green's function* involving the global Green's function and an appropriate projector. The choice of the projector determines the way in which the

coarse scales will fit the exact solution (*e.g.*, the optimal fit in the L^2 or H^1 norm). With the fine-scale Green's function in hand, an exact analytical expression for u' is possible. The downside of the Hughes and Sangalli approach is that one may not actually have the necessary expression for the global Green's function, and even in the rare cases where it can be calculated, it can be prohibitively expensive to use.

In the current work, we forgo an analytic expression for u' and attempt to approximate it *numerically*. It will be shown that an appropriate choice of a local problem with weakly enforced boundary conditions leads to a *local* approximation of the *global* Green's function expressed in terms of a given fine-scale basis. This expression may then be used in conjunction with the analytical machinery of Hughes and Sangalli to get a numerical approximation for the fine-scale Green's function, and subsequently the fine-scale solution. As a final step, the new expression for u' is inserted back into the coarse-scale equation. Numerical results will be presented that demonstrate the efficacy of this approach. Approximations to the fine-scale Green's function will be examined in detail, as well as the resulting numerical solutions to differential equations.

A simplified version of our new approach will also be presented. This new method, the parameter-free variational multiscale (PVMS) method, allows for the rapid solution of a fine-scale problem at the element level, resulting in an expression for u' that is free of the classical stabilization parameters. The method will be seen to extend naturally to many different settings.

Chapter 2

Overview of the Isogeometric Analysis Framework

Our current implementation of the isogeometric analysis concept is based on Non-Uniform Rational B-Splines (NURBS). This chapter will begin with an in depth discussion of the NURBS functions and their usage in representing various geometries comprised of a single NURBS patch. A discussion of NURBS as a basis for analysis will follow, along with a comparison with standard Finite Element Analysis (FEA). Lastly, the extension to multiple patches will be discussed, as well as one approach to local refinement.

2.1 B-splines and NURBS

2.1.1 Knot vectors

NURBS are built from B-splines and so a discussion of B-splines is a natural starting point for their investigation. Unlike in standard FEA, the B-spline parametric space is local to "patches" rather than elements. Patches play the role of ***subdomains*** within which element types and material models are assumed to be uniform. Many simple domains can be represented by a single patch.

Note that the distinction between "elements" and "patches" may be thought of in two different ways. In [42] and [43], the patches themselves are referred to as elements. This is not unreasonable as the parametric space is local to patches and a finite element code must include a loop over the patches during assembly. As mentioned previously, we take the alternate view that patches are subdomains comprised of many elements, namely the "knot spans." This latter view seems more appropriate as, in our current code, numerical quadrature is being carried out at the knot span level. Furthermore, in the case of B-splines, the functions are piecewise polynomials where the different "pieces" join along knot lines. In this way the functions are C^∞ within an element. Lastly, surpris-

ingly complicated domains may be described by a single patch (*e.g.*, all of the numerical examples in [38]). Describing such domains as being comprised of one element seems inconsistent with the traditional notion of what an element is.

A ***knot vector*** in one dimension is a set of coordinates in the parametric space, written $\Xi = \{\xi_1, \xi_2, ..., \xi_{n+p+1}\}$, where $\xi_i \in \mathbb{R}$ is the i^{th} ***knot***, i is the knot index, $i = 1, 2, ..., n + p + 1$, p is the polynomial order, and n is the number of basis functions which comprise the B-spline. The knots partition the parameter space into elements. Element boundaries in the physical space are simply the images of knot lines under the B-Spline mapping, as shown in Figure 2.1.

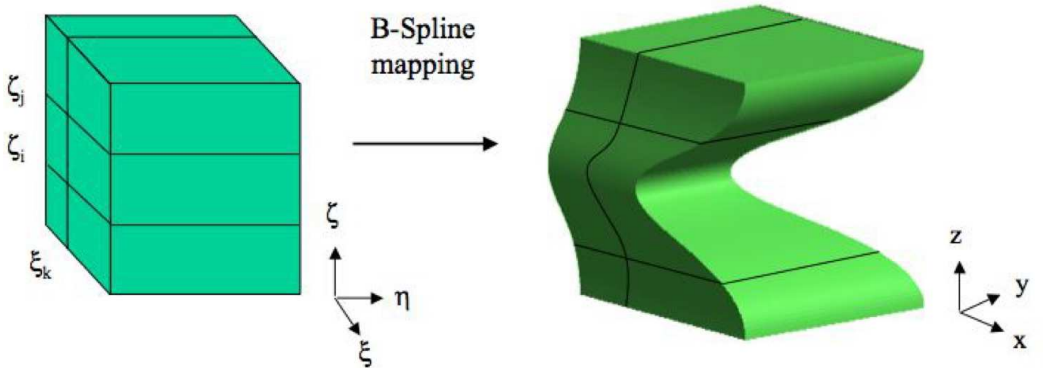


Figure 2.1: The parametric space is local to "patches" rather than elements. The knots partition the patch into elements.

Knot vectors may be ***uniform*** if the knots are equally spaced in the parametric space, or if they are unequally spaced, they are ***non-uniform***. Knot values may be repeated, that is, more than one knot may take on the same value. The multiplicities of knot values have important implications for the properties of the basis. A knot vector is said to be ***open*** if its first and last knots appear $p + 1$ times. Open knot vectors are the standard in the CAD literature. In one dimension, basis functions formed from open knot vectors are interpolatory at the ends of the parametric space interval, $[\xi_1, \xi_{n+p+1}]$, and at the corners of patches in multiple dimensions, but they are not, in general, interpolatory at interior knots. This is a distinguishing feature between knots and "nodes" in finite element analysis.

2.1.2 Basis functions

B-spline basis functions are defined recursively starting with piecewise constants $(p = 0)$:

$$N_{i,0}(\xi) = \begin{cases} 1 & \text{if } \xi_i \leq \xi < \xi_{i+1}, \\ 0 & \text{otherwise.} \end{cases} \tag{2.1}$$

For $p = 1, 2, 3, ...$, they are defined by

$$N_{i,p}(\xi) = \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} N_{i,p-1}(\xi) + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-1}(\xi). \tag{2.2}$$

The results of applying (2.1) and (2.2) to a uniform knot vector are presented in Figure 2.2. For B-spline functions with $p = 0$ and $p = 1$, we have the same result as for standard piecewise constant and linear finite element functions, respectively. Quadratic B-spline basis functions, however, look different. They are each identical but shifted relative to each other. This separates them from quadratic finite element functions which are different for internal and end nodes. This "homogeneous" pattern continues as we go to higher-order B-splines. In [38], it was hypothesized that this might result in significant advantages in equation solving over finite element functions, which are quite "heterogeneous," due to the improved conditioning of the system of equations. This seems to be the case as we were able to use iterative solvers without difficulty when modeling thin shells with solid elements – a problem that often leads to ill-conditioned systems, see Section 3.1.1. The homogeneous nature of the basis has implications for the quality of the approximation as well. In the case of structural vibrations where the heterogeneity of finite element functions leads to a branching of the spectrum that degrades the accuracy of a large percentage of the computed frequencies, the homogeneity of B-Spline functions leads to dramatic improvements, see Section 3.3.

For an open, non-uniform knot vector we can get much richer behavior. An example is presented in Figure 2.3. Note that the basis functions are interpolatory at the ends of the interval and also at $\xi = 4$, the location of a repeated knot, where only C^0-continuity is attained. Elsewhere, the functions are C^1-continuous. In general, basis functions of order p have $p - m_i$ continuous derivatives across knot ξ_i, where m_i is the multiplicity of the value of ξ_i in the knot vector. When the multiplicity of a knot value is exactly p, the basis is interpolatory at that knot. When the multiplicity is $p+1$, the basis becomes discontinuous and the patch is effectively split into two separate patches.

An important property of the B-spline basis functions is that they constitute a partition of unity, that is, $\forall \xi$,

$$\sum_{i=1}^{n} N_{i,p}(\xi) = 1. \tag{2.3}$$

This is a feature they share with meshless methods. Also of note is that the support of each $N_{i,p}$ is compact and contained in the interval $[\xi_i, \xi_{i+p+1}]$. Lastly, observe that each basis function is pointwise non-negative over the entire domain, that is, $N_{i,p}(\xi) \geq 0, \forall \xi$. This means that all of the entries of a mass matrix would be positive, which has implications for developing lumped mass schemes.

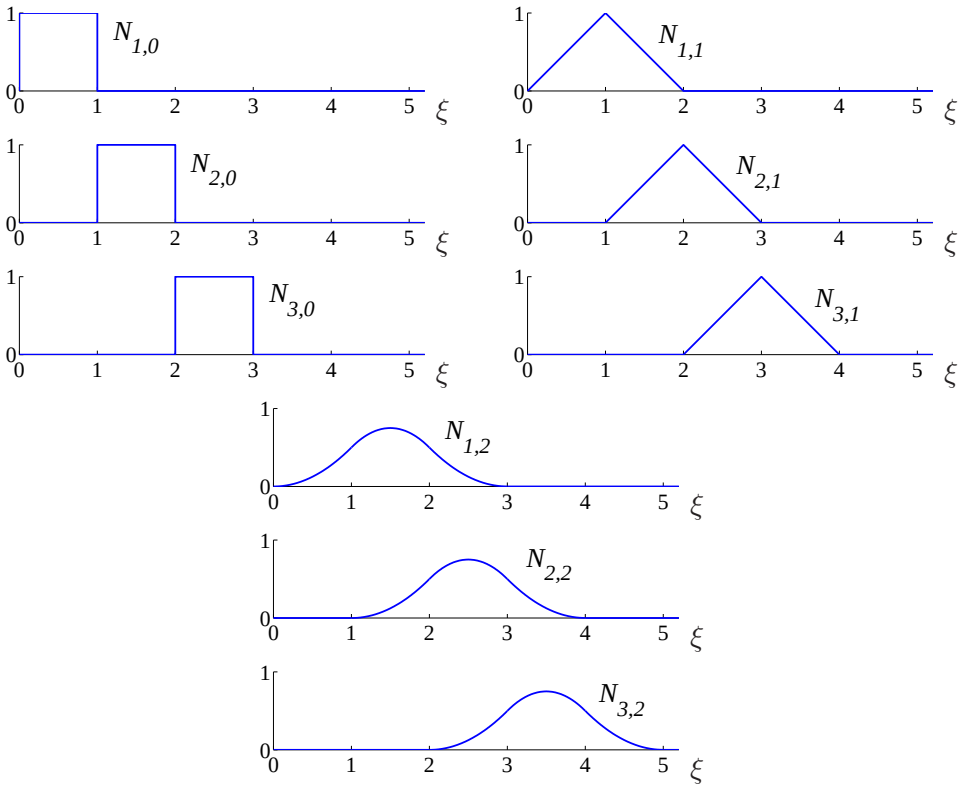


Figure 2.2: Basis functions of order $0, 1, 2$ for uniform knot vector $\Xi = \{0, 1, 2, 3, 4, ...\}$.

2.1.3 B-spline curves

B-spline curves in $\mathbb{R}^d$ are constructed by taking a linear combination of B-spline basis functions. The vector-valued coefficients of the basis functions are referred to as ***control points***. These are analogous to nodal coordinates in finite element analysis in that they are the coefficients of the basis functions, but the non-interpolatory nature of the basis does not lead to a concrete interpretation of the control point values. Piecewise linear interpolation of the control points gives the so-called ***control polygon***. Again note that, in general, control points are not interpolated by B-spline curves. Given n basis functions, $N_{i,p}, i = 1, 2, ..., n$, and corresponding control points $B_i \in \mathbb{R}^d, i = 1, 2, ..., n$, a piecewise-polynomial ***B-spline curve*** is given by

$$C(\xi) = \sum_{i=1}^{n} N_{i,p}(\xi) B_i. \tag{2.4}$$

The example shown in Figure 2.4 is built from the quadratic basis functions considered in Figure 2.3. The curve is interpolatory at the first and last control points, a general feature of a curve

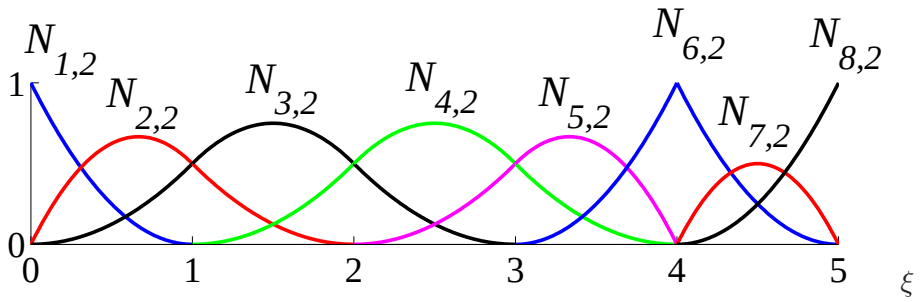
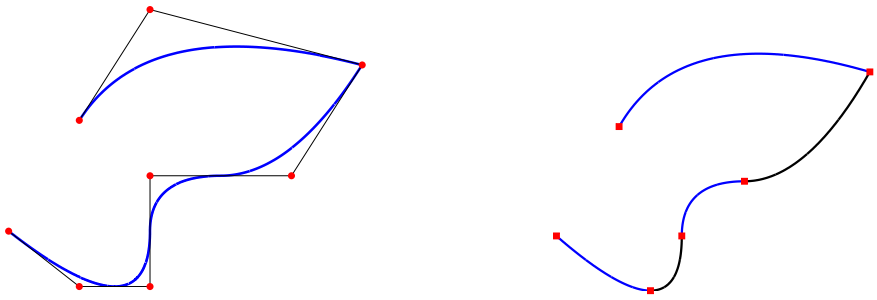


Figure 2.3: Quadratic basis functions for open, non-uniform knot vector $\Xi = \{0, 0, 0, 1, 2, 3, 4, 4, 5, 5, 5\}$.

built from an open knot vector. Note that it is also interpolatory at the sixth control point. This is due to the fact that the multiplicity of the knot $\xi = 4$ is equal to the polynomial order. Note also that the curve is tangent to the control polygon at the first, last, and sixth control points. The curve is $C^{p-1} = C^1$-continuous everywhere except at the location of the repeated knot, $\xi = 4$, where it is $C^{p-2} = C^0$-continuous. Note the difference between the control points, shown in Figure 2.4a, and the images of the knots, shown in Figure 2.4b. It is the knots, mapped into the physical space, that partition the curve into elements.



(a) Curve and control points (b) Curve and mesh denoted by knot locations

Figure 2.4: B-spline, piecewise quadratic curve in $\mathbb{R}^2$. a) Control point locations are denoted by ●'s. b) The knots, which define a mesh by partitioning the curve into elements, are denoted by ■'s. Basis functions and knot vector as in Figure 2.3.

The properties of B-spline curves follow directly from the properties of their basis functions. For example, B-spline curves have continuous derivatives of order $p - 1$ in the absence of repeated knots or control points. Repeating a knot or control point k times decreases the number of continuous derivatives by k.

Affine transformations of a B-spline curve are obtained by applying the transformations directly to the control points. This turns out to be the essential property for satisfying so-called "patch tests," as discussed in [38]. This property is referred to as ***affine covariance***.

2.1.4 *h*-refinement: Knot insertion

The mechanism for implementing *h*-refinement is ***knot insertion***.[1] Knots may be inserted without changing a curve geometrically or parametrically. Given a knot vector $\Xi = \{\xi_1, \xi_2, ..., \xi_{n+p+1}\}$, let $\bar{\Xi} = \{\bar{\xi}_1 = \xi_1, \bar{\xi}_2, ..., \bar{\xi}_{n+m+p+1} = \xi_{n+p+1}\}$ be an *extended* knot vector such that $\Xi \subset \bar{\Xi}$. The new $n+m$ basis functions are formed as before by applying (2.1) and (2.2) to the new knot vector $\bar{\Xi}$. The new $n+m$ control points, $\bar{\mathbf{B}} = \{\bar{B}_1, \bar{B}_2, ..., \bar{B}_{n+m}\}^{\mathrm{T}}$, are formed from the original control points, $\mathbf{B} = \{B_1, B_2, ..., B_n\}^{\mathrm{T}}$, by

$$\bar{\mathbf{B}} = \mathbf{T}^p \mathbf{B} \tag{2.5}$$

where

$$T_{ij}^0 = \begin{cases} 1 & \bar{\xi}_i \in [\xi_j, \xi_{j+1}) \\ 0 & \text{otherwise} \end{cases} \tag{2.6}$$

and

$$T_{ij}^{q+1} = \frac{\bar{\xi}_{i+q} - \xi_j}{\xi_{j+q} - \xi_j} T_{ij}^q + \frac{\xi_{j+q+1} - \bar{\xi}_{i+q}}{\xi_{j+q+1} - \xi_{j+1}} T_{ij+1}^q \quad \text{for} \quad q = 0, 1, 2, ..., p-1 \tag{2.7}$$

Knot values already present in the knot vector may be repeated as above but, as described in Section 2.1.2, the continuity of the *basis* will be reduced. Continuity of the *curve* is preserved by choosing the control points as in (2.5)-(2.7).

[1] Note that in the CAD literature "knot insertion" refers to inserting a single knot into a knot vector, whereas "knot refinement" refers to inserting multiple knots simultaneously. Here, we make no distinction and use "knot insertion" to refer to both cases. For an algorithm for inserting an individual knot, see [38].

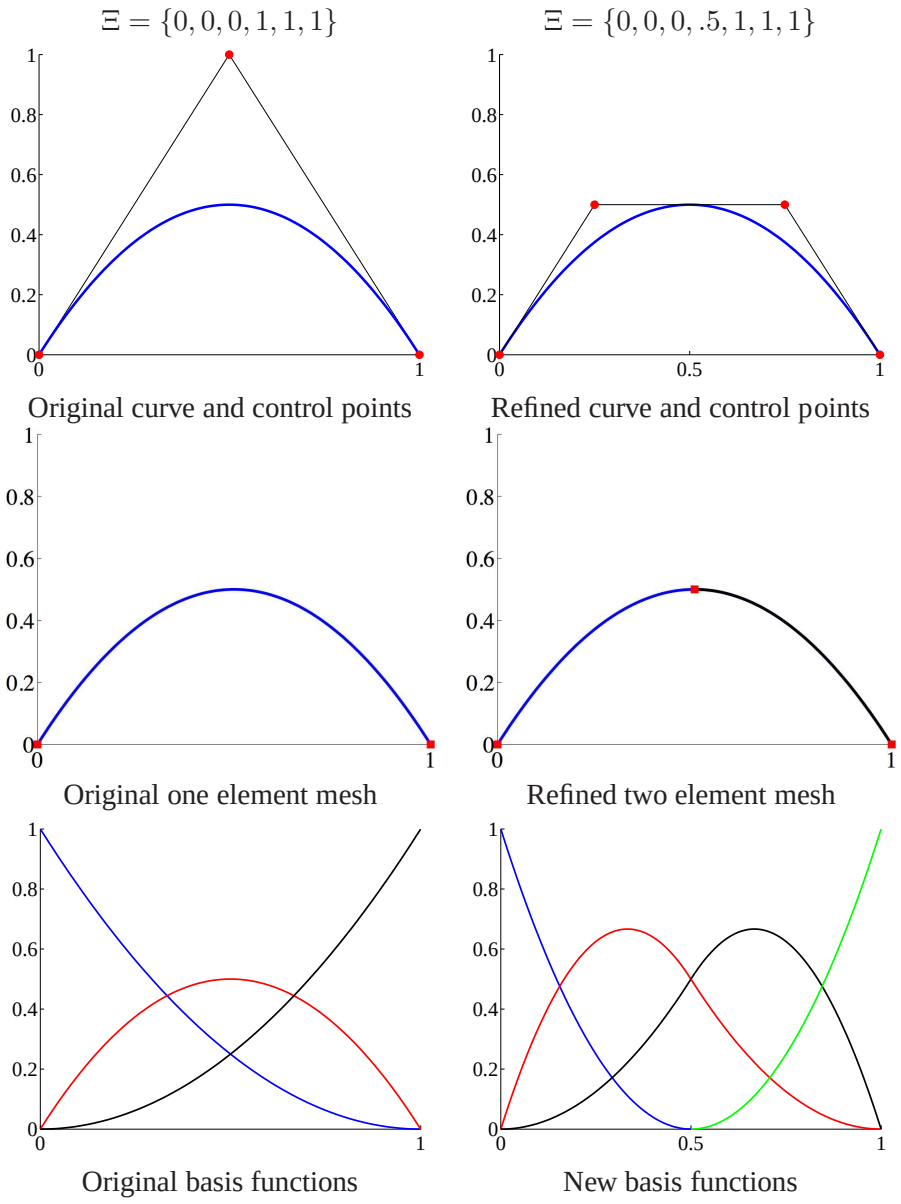


Figure 2.5: Knot insertion. Control points are denoted by ●'s. The knots, which define a mesh by partitioning the curve into elements, are denoted by ■'s.

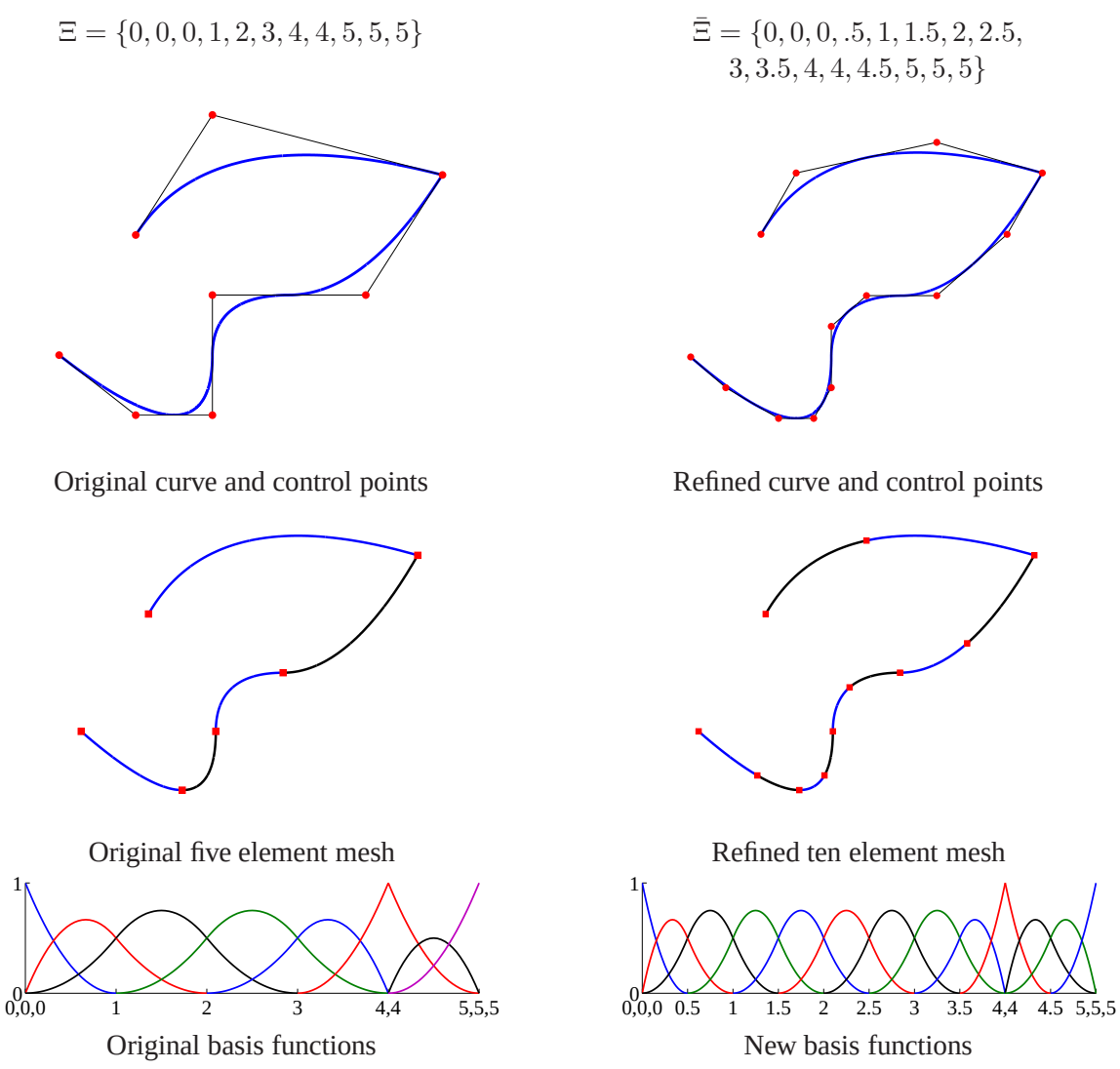


Figure 2.6: Knot insertion. Control points are denoted by ●'s. The knots, which define a mesh by partitioning the curve into elements, are denoted by ■'s. Each element has been evenly split in the parametric domain.

An example of knot insertion for a simple, one-element curve is presented in Figure 2.5. The original curve consists of quadratic B-splines. The knot vector is $\Xi = \{0, 0, 0, 1, 1, 1\}$. The control points, mesh, and basis functions of the unrefined curve are shown on the left. A new knot is inserted at $\bar{\xi} = 0.5$. The new curve, shown on the right, is geometrically and parametrically identical to the original curve, but the control points are changed, the mesh is partitioned, and the basis is richer. There is one more control point, one more element, and one more basis function. This process may be repeated to enrich the solution space by adding more basis functions of the same order while leaving the curve unchanged. Figure 2.6 shows the more advanced case of a global refinement of the curve from Figure 2.4. Insertion of new knot values has parallels with the classical *h*-refinement strategy in finite element analysis as it splits existing elements into new ones. Repeating existing knot values to decrease the continuity of the basis does not have an analogue in FEA. We will revisit this idea below.

2.1.5 *p*-refinement: Order elevation

The mechanism for implementing *p*-refinement is ***order elevation***[2]. As its name implies, the process involves raising the polynomial order of the basis functions used to represent the geometry (and the solution space, as our finite element implementation will be isoparametric). Recalling from Section 2.1.1 that the basis has $p - m_i$ continuous derivatives across element boundaries, it is clear that, when p is increased, m_i must also be increased if we are to preserve the discontinuities in the derivatives of our original curve. During order elevation, the *multiplicity* of each knot value is increased by one, but no new knot *values* are added. As with knot insertion, neither the geometry nor the parameterization are changed.

The process for order elevation begins by replicating existing knots until their multiplicity is equal to the polynomial order, thus effectively subdividing the curve into many Bézier curves by knot insertion (see Rogers [48] or Farin [25] for a discussion of Bézier curves; we may think of them as one element B-spline curves). The next step is to elevate the order of the polynomial on each of these individual segments. Lastly, excess knots are removed to combine the segments into one, order-elevated, B-spline curve. Several efficient algorithms exist which combine the steps so as to minimize the computational cost of the process. Details are omitted for the sake of brevity. For a thorough treatment, see Piegl and Tiller [46].

[2] sometimes also called "degree elevation"

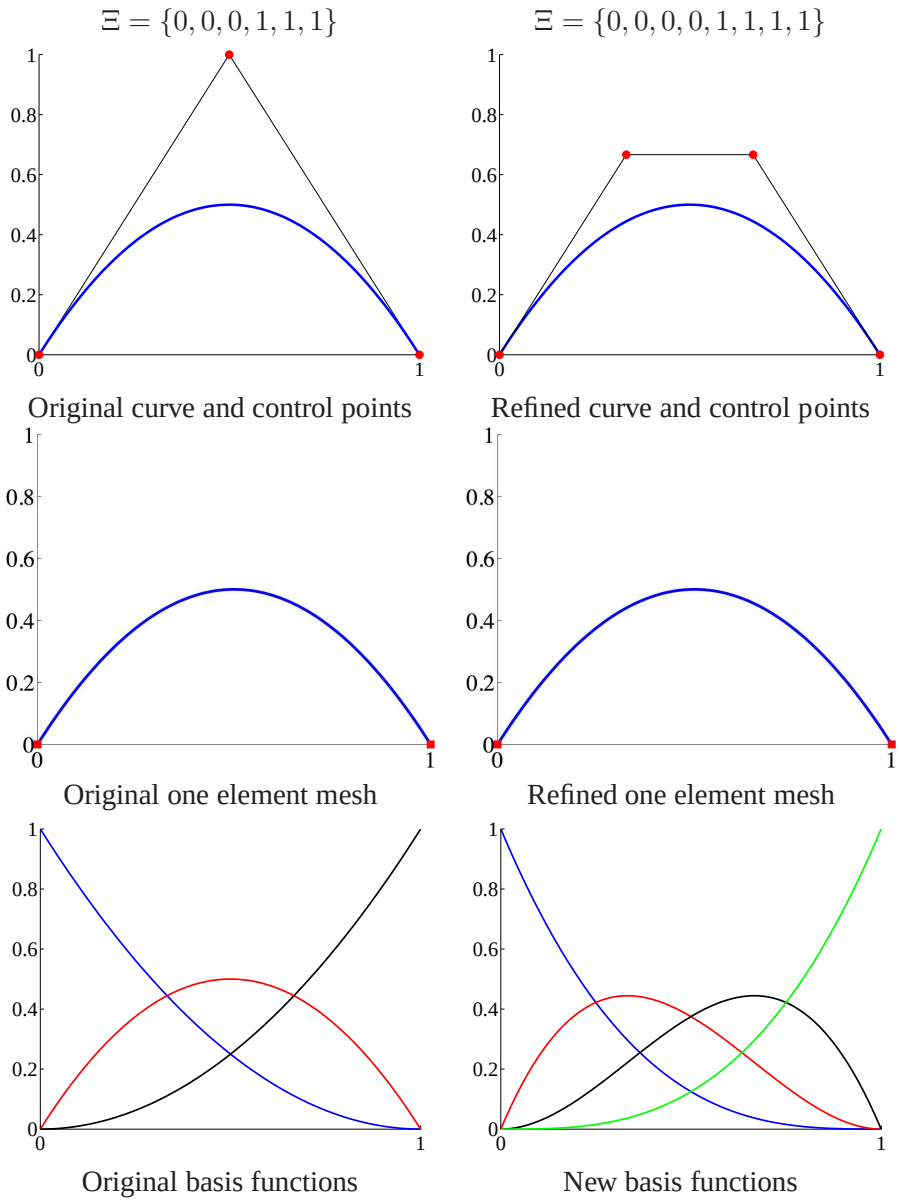


Figure 2.7: Order elevation. Control points are denoted by •'s. The knots, which define a mesh by partitioning the curve into elements, are denoted by ■'s.

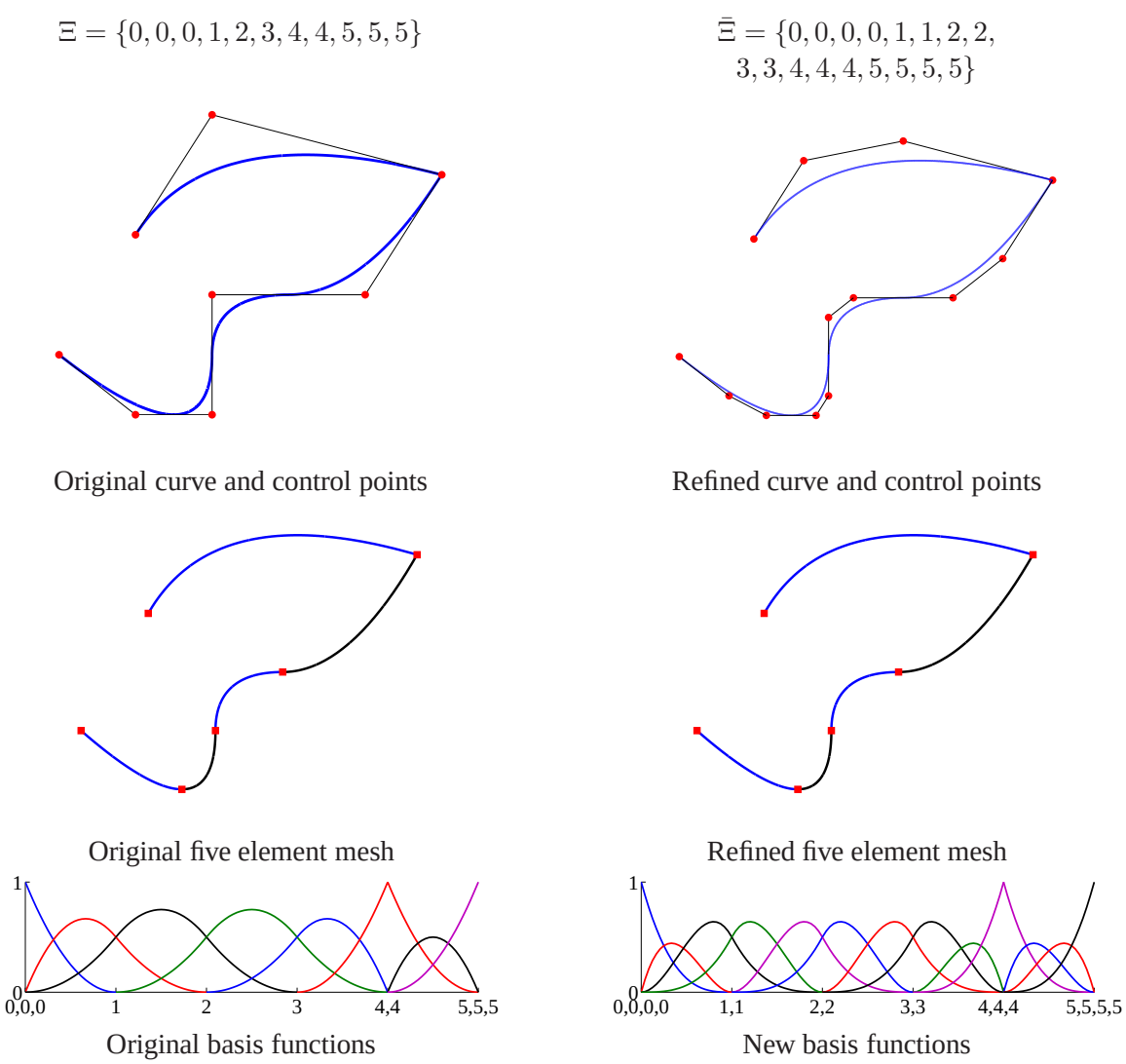


Figure 2.8: Order elevation. Control points are denoted by ●'s. The knots, which define a mesh by partitioning the curve into elements, are denoted by ■'s. Note the increased multiplicity of internal knots. This is done to preserve discontinuities in the derivatives of the curve.

An example of order elevation for a one-element curve is depicted in Figure 2.7. The original control points, mesh, and quadratic basis functions, shown on the left, are the same as considered in Figure 2.5. This time the *multiplicity* of the knots is increased by one but, as stated above, no new knot *values* are added. For this simple case, the numbers of control points and basis functions each increase by one. The locations of the control points change, but the elevated curve is geometrically and parametrically identical to the original curve. There are now four cubic basis functions. Figure 2.8 shows this process on the more complex example considered in Figure 2.6. The multiplicities of the knots have been increased but no new elements created. Note that the locations of control points for these order elevated curves are different than those in the *h*-refinement examples (cf. Figs. 2.5 and 2.6).

2.1.6 *k*-refinement: Higher order *and* higher continuity

As we have seen, the two primitive refinement operations for B-splines are knot insertion and order elevation. Knot insertion is similar to *h*-refinement, but for it to be a perfect analogue each new knot value would have to be inserted with multiplicity $m_i = p$ to ensure a C^0 basis everywhere. Similarly, if we begin with a mesh in which all of the functions are already C^0 across element boundaries, order elevation coincides exactly with our traditional notion of *p*-refinement. Knot insertion and order elevation, however, provide us with more to work with than do the two standard notions of refinement.

As mentioned above, we can insert new knot values with multiplicities of one to define new elements across whose boundaries functions will be C^{p-1}. We can also repeat existing knot values to lower the continuity of the basis across existing element boundaries. This makes knot insertion a more flexible process than simple *h*-refinement. Similarly, we have a more flexible higher-order refinement as well. It stems from the fact that the processes of order elevation and knot insertion do not commute. If a unique knot value, $\bar{\xi}$, is inserted between two distinct knot values in a curve of order p, the number of continuous derivatives of the basis functions at $\bar{\xi}$ is $p-1$. As described above, if we subsequently elevate the order to q, the multiplicity of every distinct knot value (including the knot just inserted) is increased so that discontinuities in the p^{th} derivative of the basis are preserved. That is, the basis still has $p-1$ continuous derivatives at $\bar{\xi}$, although the order is now q. If, instead, we elevated the order of the original, coarsest curve to q and only then inserted the unique knot value $\bar{\xi}$, the basis would have $q-1$ continuous derivatives at $\bar{\xi}$. We refer to this latter procedure as ***k-refinement***. We know of no analogous practice in standard finite element analysis.

Remark 2.1.1. *This notion of k-refinement is not the same as the "k-convergence" described in [42] in which the position of the knots is altered. It bears more in common with the "k-version finite element method" of [53, 54] in that k refers to continuity, but the motivations are different. The*

increased continuity in [53] is required so that a least-squares finite element approach is possible. Such an approach requires that the solution space have the same number of continuous derivatives as found in the highest order derivative of the differential operator. Our motivations for using basis functions of higher continuity are efficiency and robustness of the solution space in a classical Galerkin finite element formulation of the problem.

This concept of *k*-refinement is important because isogeometric analysis is fundamentally a higher-order approach. While linear finite elements can be represented within a NURBS context, it takes quadratic-level NURBS to represent conic sections – one of the key features of the approach. This is potentially a superior approach to high-precision analysis than *p*-refinement. In traditional *p*-refinement there is a very inhomogeneous structure to arrays due to the different basis functions associated with surface, edge, vertex and interior nodes. In addition, there is a proliferation in the number of nodes because C^0-continuity is maintained in the refinement process. In *k*-refinement, there is a homogeneous structure within patches and growth in the number of control variables is limited. Let us reemphasize that an "element" in one dimension is the span between two *distinct* knot values. The number of elements in a curve will then be the number of non-zero knot spans in the knot vector (e.g., the domain associated with the knot vector $\Xi = \{0, 0, 0, 1, 2, 3, 3, 4, 4, 4\}$ consists of four elements).

Consider the classical *p*-refinement process (see Figures 2.9b and 2.10a). Assume the initial domain consists of one element and $p + 1$ basis functions (assuming an open knot vector), which we then refine by inserting new knot values until we have $n - p$ elements and n basis functions, all C^{p-1}. We then perform order elevation, maintaining continuity at the $p - 1$ level. This requires replicating each distinct knot value, adding a basis function in each element and so increasing the total number of basis functions by $n-p$ to $2n-p$. After a total of r order elevations of this type, we have $(r + 1)n - rp$ basis functions, where p is still the order of our original basis functions. This is seen to be a large number of functions when one considers that in most cases of practical interest the number of elements will be quite a bit larger than the order of the basis. For comparison, consider beginning with the same one element domain and proceeding by *k*-refinement, as in Figures 2.9c and 2.10b. That is, order elevate r times adding only *one* basis function at each refinement, then insert knots until we have $n - p$ elements as before. The final number of basis functions is $n + r$, each having $r + p - 1$ continuity. This amounts to an enormous savings as $n + r$ is considerably smaller than $(r+1)n-rp$. Additionally, keep in mind that in d dimensions these numbers are raised to the d power. Graphical comparisons are shown in Figures 2.11-2.13. Note that the mesh, defined by the knot *locations*, is fixed and is the same for *p*- and *k*-refinements.

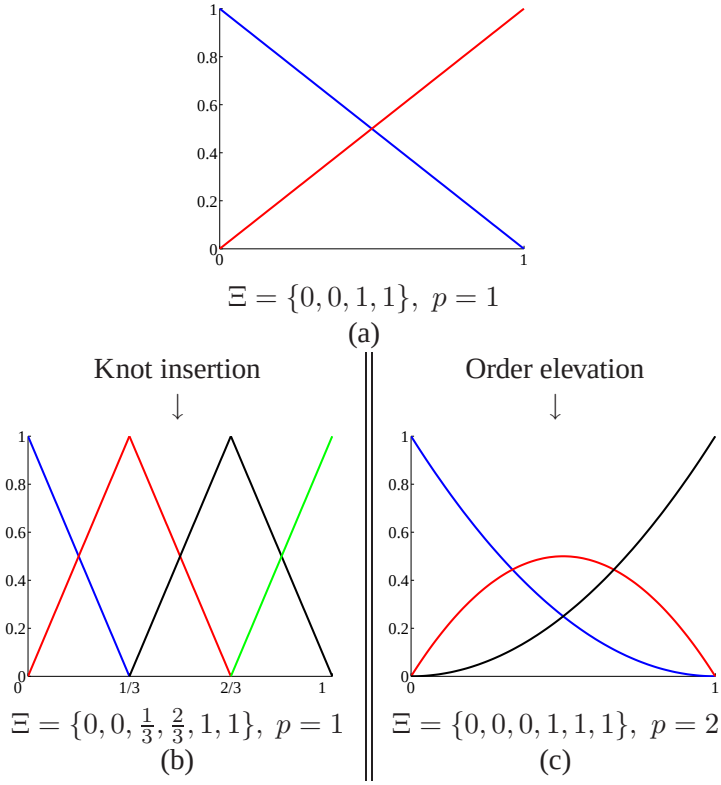
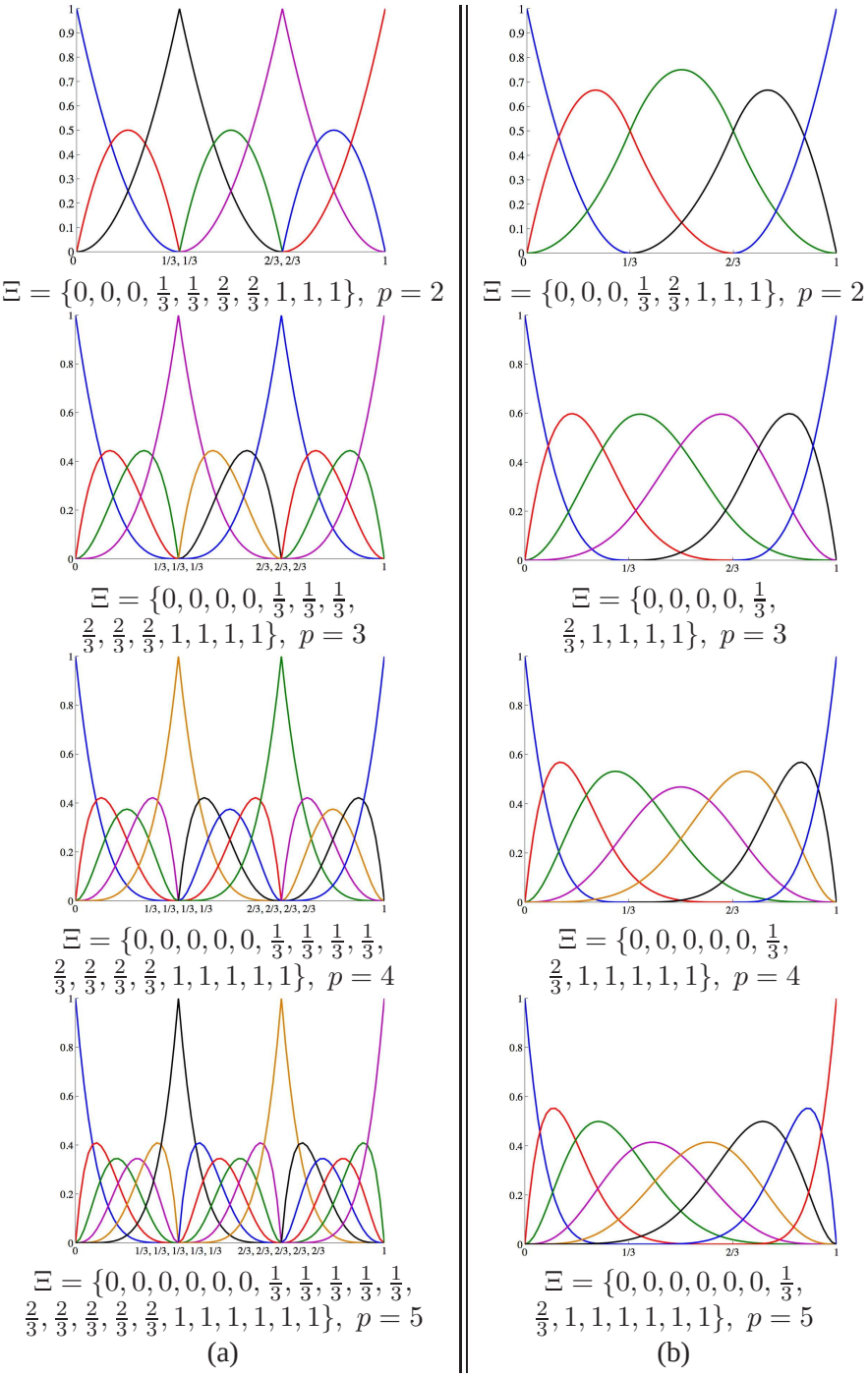


Figure 2.9: When refining a coarse, low-order mesh to create a fine, higher-order mesh, one may choose between a p- or k-refinement strategy. Here we see the initial step for each case. (a) Base case of one linear element. (b) Classic *p*-refinement approach: knot insertion is performed first to create many low-order elements. Subsequent order elevation will preserve the C^0 continuity across element boundaries. c) New *k*-refinement approach: order elevation is performed on the coarsest discretization. Subsequent knot insertion will result a basis which is C^{p-1} across the newly created element boundaries. See the results of p- and k-refinement for several different polynomial orders in Figure 2.10.



$\Xi = \{0, 0, 0, \frac{1}{3}, \frac{1}{3}, \frac{2}{3}, \frac{2}{3}, 1, 1, 1\},\ p = 2$

$\Xi = \{0, 0, 0, \frac{1}{3}, \frac{2}{3}, 1, 1, 1\},\ p = 2$

$\Xi = \{0, 0, 0, 0, \frac{1}{3}, \frac{1}{3}, \frac{1}{3}, \frac{2}{3}, \frac{2}{3}, \frac{2}{3}, 1, 1, 1, 1\},\ p = 3$

$\Xi = \{0, 0, 0, 0, \frac{1}{3}, \frac{2}{3}, 1, 1, 1, 1\},\ p = 3$

$\Xi = \{0, 0, 0, 0, 0, \frac{1}{3}, \frac{1}{3}, \frac{1}{3}, \frac{1}{3}, \frac{2}{3}, \frac{2}{3}, \frac{2}{3}, \frac{2}{3}, 1, 1, 1, 1, 1\},\ p = 4$

$\Xi = \{0, 0, 0, 0, 0, \frac{1}{3}, \frac{2}{3}, 1, 1, 1, 1, 1\},\ p = 4$

$\Xi = \{0, 0, 0, 0, 0, 0, \frac{1}{3}, \frac{1}{3}, \frac{1}{3}, \frac{1}{3}, \frac{1}{3}, \frac{2}{3}, \frac{2}{3}, \frac{2}{3}, \frac{2}{3}, \frac{2}{3}, 1, 1, 1, 1, 1, 1\},\ p = 5$

(a)

$\Xi = \{0, 0, 0, 0, 0, 0, \frac{1}{3}, \frac{2}{3}, 1, 1, 1, 1, 1, 1\},\ p = 5$

(b)

Figure 2.10: Three element, higher-order meshes for p- and k-refinement. a) The p-refinement approach results in many functions that are C^0 across element boundaries. b) In comparison, k-refinement results in a much smaller number of functions, each of which is C^{p-1} across element boundaries.

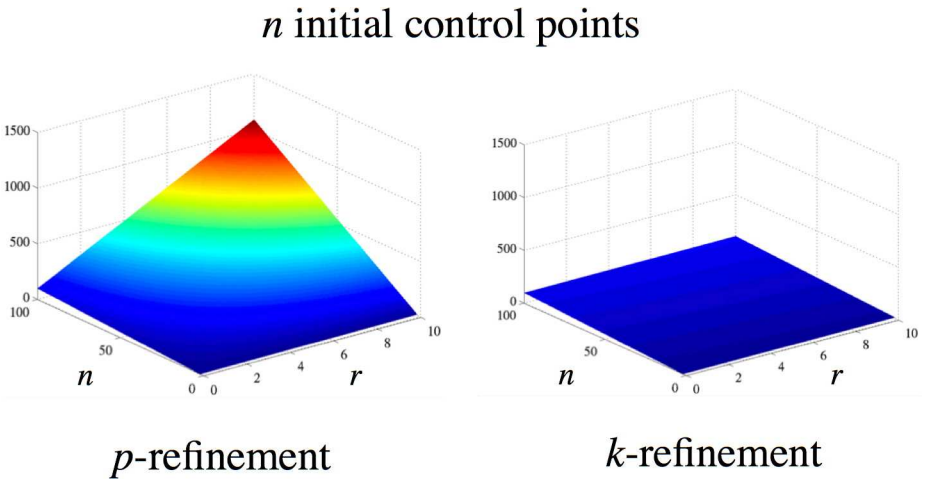


Figure 2.11: Comparison of control variable growth in one dimension.

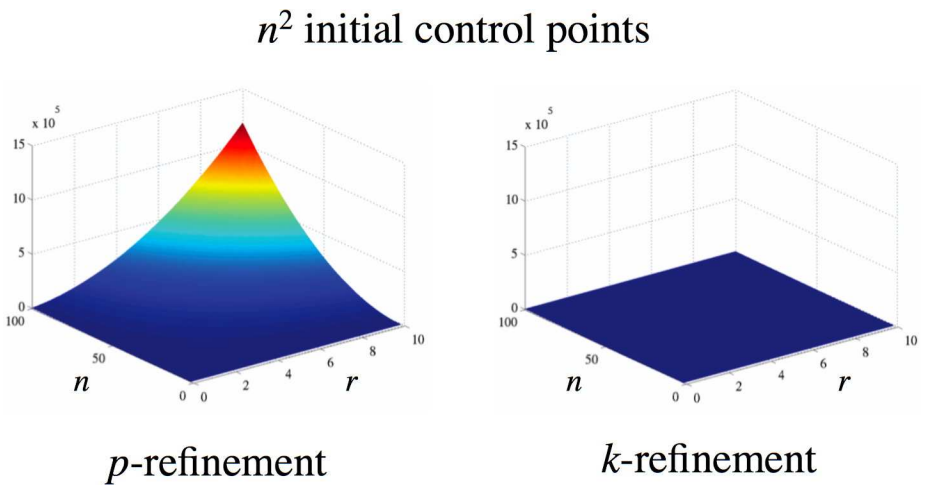


Figure 2.12: Comparison of control variable growth in two dimensions.

There are advantages beyond the slower growth in degrees-of-freedom. The smoother functions may lead to more accurate physical quantities, such as strains and stresses. Perhaps more striking, the increased homogeneity of the functions along with their high continuity lead to dramatic improvements over standard finite elements for structural vibration analysis.

It is important to note that "pure" *k*-refinement, where all functions maintain C^{p-1} continuity across element boundaries, is only possible if the coarsest mesh is comprised of one element. If the initial mesh places constraints on the continuity across certain element boundaries, these constraints will exist on all meshes. In general, though some such constraints will exist, the number

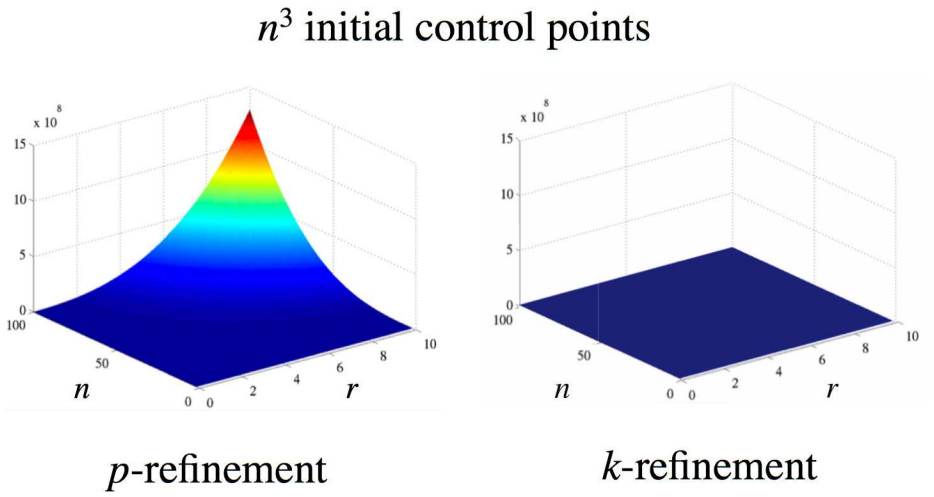


Figure 2.13: Comparison of control variable growth in three dimensions.

of elements desired for analysis will be much higher than the number needed for modeling the geometry. Refinements may be performed such that the functions have $p-1$ continuous derivatives across these new element boundaries and the benefits of *k*-refinement will still be significant.

The *hpk*-refinement space

As we have shown, knot insertion and order elevation are the primitive operations by which classical h- and p-refinements, as well as the new k-refinement, can be implemented. Recognizing their flexibility as compared with classical refinement procedures makes feasible the notion of an hpk-refinement space. Recalling that B-spline curves may have no more than $p-1$ continuous derivatives across an element boundary, the set of possible refinements may be characterized as in Figure 2.14. Pure k-refinement keeps h fixed but increases the continuity along with the polynomial order, as in Figure 2.15. Pure p-refinement increases the polynomial order while the basis remains C^0, as in Figure 2.16. Increasing the multiplicity of existing knot values decreases the continuity without introducing new elements, as in Figure 2.17. Inserting new knot values with a multiplicity of p results in classical h-refinement, whereby new elements are introduced that have C^0 boundaries, shown in Figure 2.18. Inserting new knot values with a multiplicity of 1 decreases h without decreasing the minimum continuity already found in the mesh, as in Figure 2.19. Considering all of the aforementioned techniques results in a multitude of refinement options beyond simple h-, p- and k-refinement, see Figure 2.20.

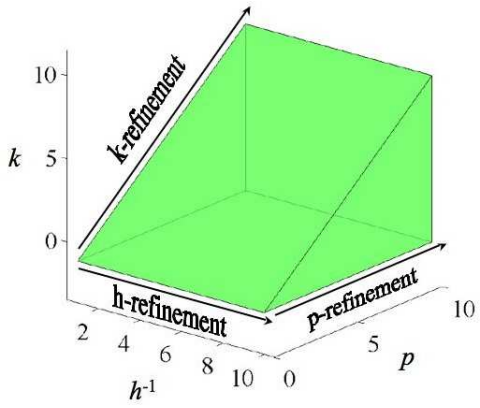


Figure 2.14: The hpk-space. The set of all allowable refinements is contained in the region shown in green. Note that this region extends in the direction of the arrows.

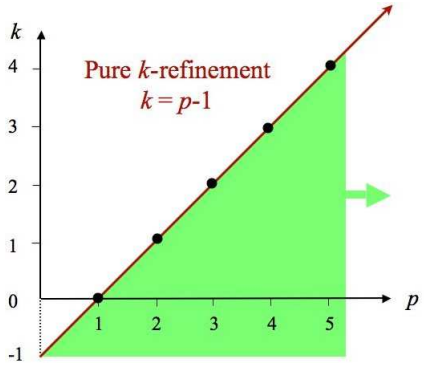


Figure 2.15: The hpk-space. In pure k-refinement, the locations of the element boundaries (and thus element size, h) are fixed. As the polynomial order, p, is increased, the continuity of the functions across element boundaries, k, is increased such that $k = p - 1$ at all levels of refinement.

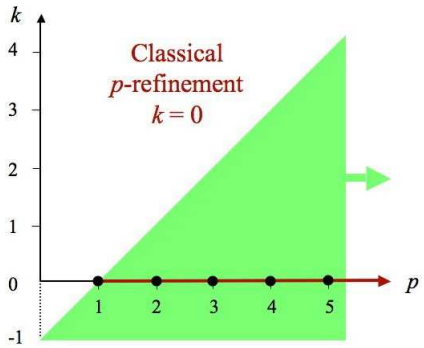


Figure 2.16: The hpk-space. In pure p-refinement, the locations of the element boundaries (and thus element size, h) are fixed. As the polynomial order, p, is increased, the continuity of the functions across element boundaries is fixed at $k = 0$ for all levels of refinement.

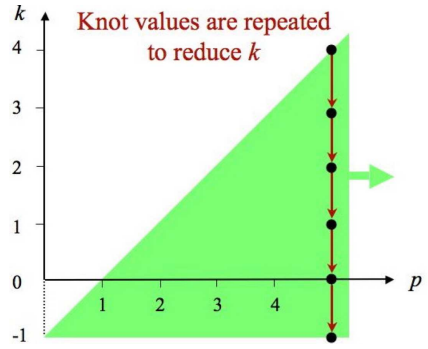


Figure 2.17: The hpk-space. Repetition of existing knot values decreases the continuity across the corresponding element boundary without creating new elements or changing the polynomial order. The basis has $p - m_i$ continuous derivatives across knot ξ_i, where m_i is the multiplicity of that knot value.

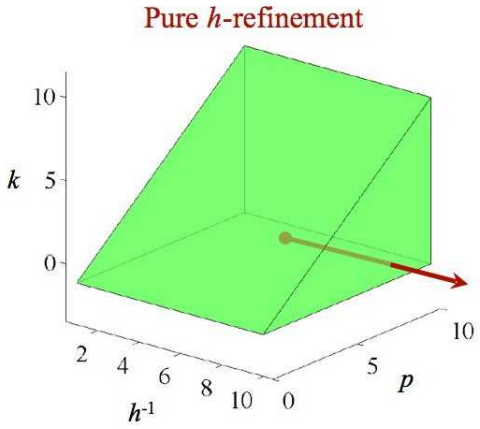


Figure 2.18: The hpk-space. If we insert new knot values with multiplicity of p, new elements are created and the basis remains C^0 across all element boundaries. In this way classical h-refinement is exactly replicated.

2.1.7 B-spline surfaces

Given a ***control net*** $\{B_{i,j}\}, i = 1, 2, ..., n, j = 1, 2, ..., m$, and knot vectors $\Xi = \{\xi_1, \xi_2, ..., \xi_{n+p+1}\}$, and $\mathcal{H} = \{\eta_1, \eta_2, ..., \eta_{m+q+1}\}$, a tensor product B-spline surface is defined by

$$S(\xi, \eta) = \sum_{i=1}^{n} \sum_{j=1}^{m} N_{i,p}(\xi) M_{j,q}(\eta) B_{i,j} \tag{2.8}$$

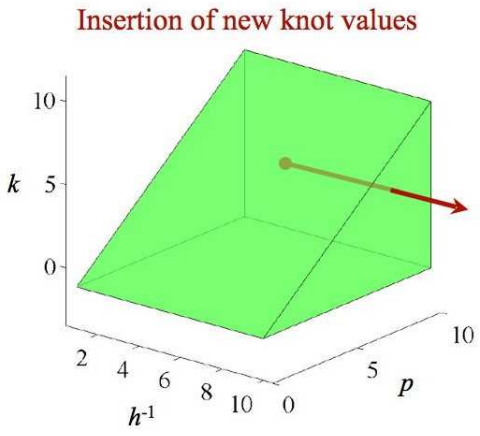


Figure 2.19: The hpk-space. Insertion of new knot values with a multiplicity of 1 results in a splitting of elements, and thus a decrease in h (shown in the figure as an increase in h^{-1}). The basis has $p-1$ continuous derivatives across these new element boundaries, and so the (possibly lower) minimum continuity already existing in the mesh is unchanged, as is the polynomial order.

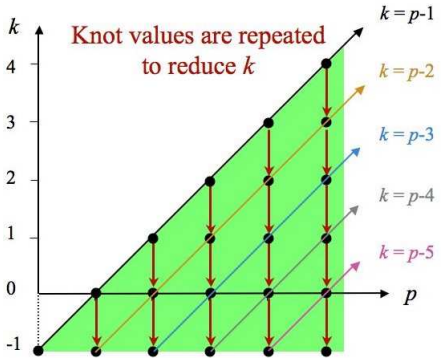


Figure 2.20: The hpk-space. Combining knot insertion and order elevation in various permutations allows us to traverse the entire allowable refinement space.

where $N_{i,p}$ and $M_{j,q}$ are basis functions of B-spline curves. For purposes of numerically integrating arrays constructed from B-splines, "elements" are taken to be knot spans, namely, $[\xi_i, \xi_{i+1}] \times [\eta_j, \eta_{j+1}]$. See Figure 2.21 for an illustration of a standard bi-unit parent element and its image in physical space. Integrals are pulled back to the parent element by the classical change-of-variables formula and standard Gaussian quadrature rules are employed; see Hughes [32], Chapter 3.

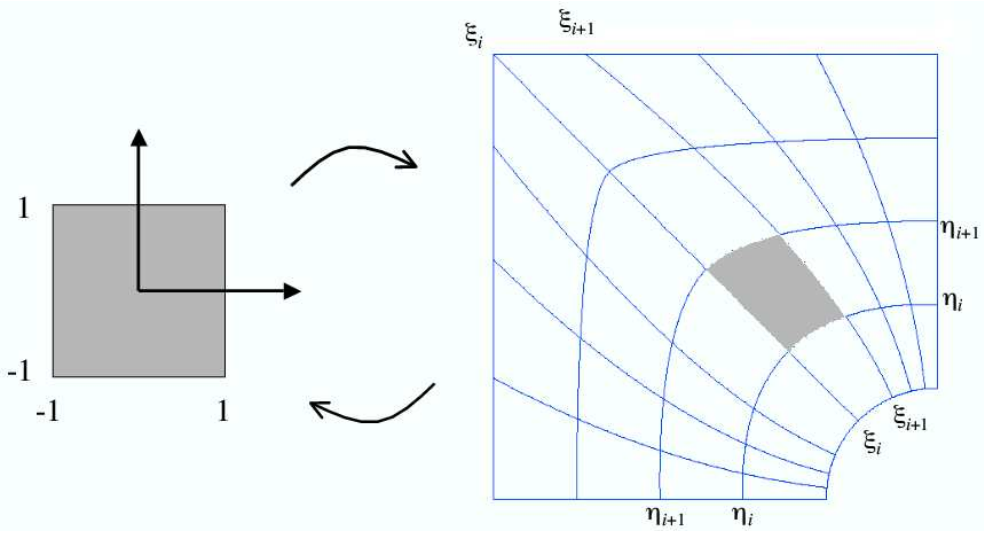


Figure 2.21: Elements as knot spans.

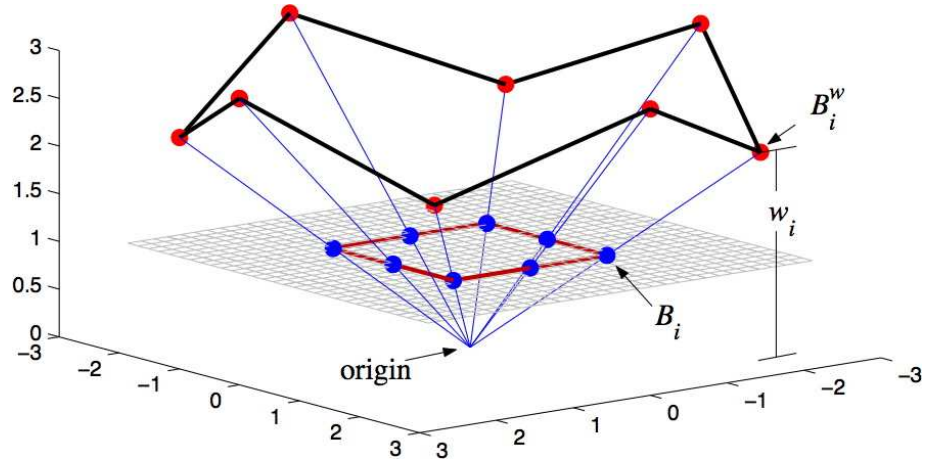
2.1.8 B-spline solids

Tensor product B-spline solids are defined in analogous fashion to B-spline surfaces. Given a ***control lattice*** $\{B_{i,j,k}\}, i = 1, 2, ..., n, j = 1, 2, ..., m, k = 1, 2, ..., l$, and knot vectors $\Xi = \{\xi_1, \xi_2, ..., \xi_{n+p+1}\}$, $\mathcal{H} = \{\eta_1, \eta_2, ..., \eta_{m+q+1}\}$, and $\mathcal{Z} = \{\zeta_1, \zeta_2, ..., \zeta_{l+r+1}\}$, a B-spline solid is defined by

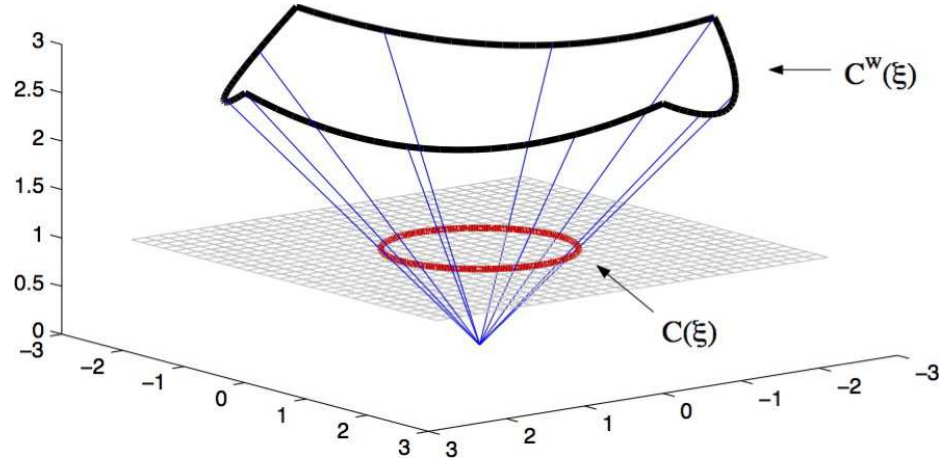
$$S(\xi, \eta, \zeta) = \sum_{i=1}^{n} \sum_{j=1}^{m} \sum_{k=1}^{l} N_{i,p}(\xi) M_{j,q}(\eta) L_{k,r}(\zeta) B_{i,j,k} \tag{2.9}$$

2.1.9 Rational B-splines

As described in the beginning of this Chapter, NURBS are formed from B-splines. Specifically, NURBS entities in $\mathbb{R}^d$ can be obtained by projective transformations of B-spline entities in $\mathbb{R}^{d+1}$, in particular, conic sections, such as circles and ellipses, can be *exactly* constructed by projective transformations of piecewise rational quadratic curves. This is illustrated in Figure 2.22 in which a



(a) Control polygons



(b) Curves

Figure 2.22: Circle in $\mathbb{R}^2$ constructed by projective transformation of piecewise quadratic B-spline in $\mathbb{R}^3$. (a) Projective transformation of "projective control points" yields control points. Weight w_i is the z-component of B_i^w. (b) Projective transformation of B-spline curve $C^w(\xi)$ yields NURBS curve $C(\xi)$.

circle in $\mathbb{R}^2$ is constructed from a piecewise quadratic B-spline curve in $\mathbb{R}^3$. The projective transformation of a B-spline curve yields a rational polynomial of the form $C_R(\xi) = f(\xi)/g(\xi)$, where f and g are piecewise polynomials. The construction of a rational B-spline curve in $\mathbb{R}^d$ proceeds as follows. Let $\{B_i^w\}$ be a set of control points for a B-spline curve in $\mathbb{R}^{d+1}$ with knot vector Ξ. These are referred to as the "projective control points" for the desired NURBS curve in $\mathbb{R}^d$. The control points in $\mathbb{R}^d$ are derived from the projective control points by the following relations:

$$(B_i)_j = (B_i^w)_j / w_i, \quad j = 1, ..., d \tag{2.10}$$

$$w_i = (B_i^w)_{d+1} \tag{2.11}$$

where $(B_i)_j$ is the j^{th} component of the vector B_i, etc. and w_i is referred to as the i^{th} ***weight***. In Figure 2.22a, the weights are the vertical coordinates of the control points defining the piecewise quadratic B-spline curve in $\mathbb{R}^3$. The rational basis functions and NURBS curve are given by

$$R_i^p(\xi) = \frac{N_{i,p}(\xi) w_i}{\sum_{\hat{i}=1}^{n} N_{\hat{i},p}(\xi) w_{\hat{i}}} \tag{2.12}$$

$$C(\xi) = \sum_{i=1}^{n} R_i^p(\xi) B_i \tag{2.13}$$

Rational surfaces and solids are defined analogously in terms of the rational basis functions

$$R_{i,j}^{p,q}(\xi, \eta) = \frac{N_{i,p}(\xi) M_{j,q}(\eta) w_{i,j}}{\sum_{\hat{i}=1}^{n} \sum_{\hat{j}=1}^{m} N_{\hat{i},p}(\xi) M_{\hat{j},q}(\eta) w_{\hat{i},\hat{j}}} \tag{2.14}$$

$$R_{i,j,k}^{p,q,r}(\xi, \eta, \zeta) = \frac{N_{i,p}(\xi) M_{j,q}(\eta) L_{k,r}(\zeta) w_{i,j,k}}{\sum_{\hat{i}=1}^{n} \sum_{\hat{j}=1}^{m} \sum_{\hat{k}=1}^{l} N_{\hat{i},p}(\xi) M_{\hat{j},q}(\eta) L_{\hat{k},r}(\zeta) w_{\hat{i},\hat{j},\hat{k}}} \tag{2.15}$$

The powerful thing about the construction of the NURBS basis functions is that, as NURBS in $\mathbb{R}^d$ are B-Splines in $\mathbb{R}^{d+1}$, all of the refinement techniques we have discussed are applied to NURBS by operating directly on those higher dimensional B-Splines. The NURBS basis functions also form a partition of unity. The continuity and support of NURBS are the same as for B-splines. Affine transformations in physical space are still obtained by applying the transformation to the control points, that is, NURBS possess the property of affine covariance.

It is interesting to note that, if the weights are all equal, NURBS become B-splines (i.e., piecewise polynomials). As *h*-refinement progresses, the standard deviation of the weights within an element decreases, so *locally* the basis becomes polynomial. This is one of the reasons why standard Gaussian integration rules have been observed to be adequate for the rational functions.

Remark 2.1.2. *Careful integration of NURBS functions warrants proper investigation. Our experience has been that, though the standard Gaussian rules may be insufficient for NURBS on the*

coarsest of meshes, by the time the mesh is fine enough to get a reasonable solution to the equations, the elements are small enough (and the basis close enough to polynomial) that the Gauss rules are sufficient and no benefit is seen from increasing the quadrature rule. In comparing with standard FEA, it is worth recalling that even for a polynomial basis, Gaussian integration is not exact over curved elements.

2.2 NURBS as a basis for analysis

As discussed in Chapter 1, analysis with NURBS still begins with mesh generation. The product of knot vectors (given in three dimensions, by $\Xi \times \mathcal{H} \times \mathcal{Z}$) defines the parametric domain. Basis functions are generated as described above. The control points multiply these basis functions to define the NURBS mapping of the parameter space into the physical domain. The images of the knot lines under this mapping are the element boundaries (recall Figure 2.1). As we have seen, the support of each basis function is compact, consisting of the union of a small number of these elements.

To represent solutions to systems of partial differential equations (PDEs), we invoke the isoparametric concept. That is, the fields in question (e.g., displacement, velocity, temperature, etc.) are represented in terms of the same basis functions as the geometry. The coefficients of the basis functions are the degrees-of-freedom, or ***control variables***. Like the control points, the control variables are not, in general, interpolated. This means that their actual values do not have a specific interpretation but, when paired with the NURBS basis functions, they describe a solution field with all of the continuity properties of that underlying basis. Mesh refinement strategies are developed from a combination of knot insertion and order elevation techniques. These enable analogues of classical *h*-refinement and *p*-refinement methods, the new possibility of *k*-refinement, and many things in between, all as described above.

The overall structure of an isogeometric analysis code does not differ greatly from an FEA code. Arrays constructed from isoparametric NURBS elements can be assembled into global arrays in the same way as finite elements; see Hughes [32], Chapter 2. When multiple patches are used to describe the domain (see Section 2.3), a loop over patches is needed outside of the loop over elements. Compatibility of the NURBS patches is discussed in the next section.

The easiest way to set Dirichlet boundary conditions is to apply them to the control variables. In the case of homogeneous Dirichlet conditions, this results in their exact, pointwise satisfaction. In the case of inhomogeneous Dirichlet conditions, the boundary values must be approximated by functions lying within the NURBS space. This amounts to "strong" but approximate satisfaction of the boundary conditions. Constraint equations can also be written to ensure strong exact, interpolated, or least-squares best-fit satisfaction of boundary conditions. An alternative formulation of

Dirichlet conditions can be based on "weak" satisfaction (see Bazilevs and Hughes [7]), a standard feature of the discontinuous Galerkin method and a concept we will return to in Chapters 5 and 6. Given the variety of possibilities, Dirichlet boundary conditions need to be researched more thoroughly to determine optimal strategies. Neumann boundary conditions are satisfied naturally, in precisely the same way as in standard finite element formulations; see Hughes [32] Chapters 1 and 2.

It is well known that typical finite element interpolation functions oscillate in attempting to fit discontinuous data. An example is illustrated in Figure 2.23a where Lagrange polynomials of orders three, five, and seven interpolate a discontinuity represented by eight data points in $\mathbb{R}^2$. Note that as the order is increased, the amplitude of the oscillations increases. This is sometimes referred to as ***Gibbs phenomena***. NURBS behave very differently when the data are viewed as control points. In Figure 2.23b the NURBS curves are monotone, illustrating the ***variation diminishing*** property of NURBS (see Rogers [48], Chapter 3). In [38], this property was hypothesized to have advantages in representing sharp layers. It is hoped that it can be exploited in some way to increase the robustness of a solver in the presence of shocks, though doing so in a manner that is anything but *ad hoc* may prove difficult.

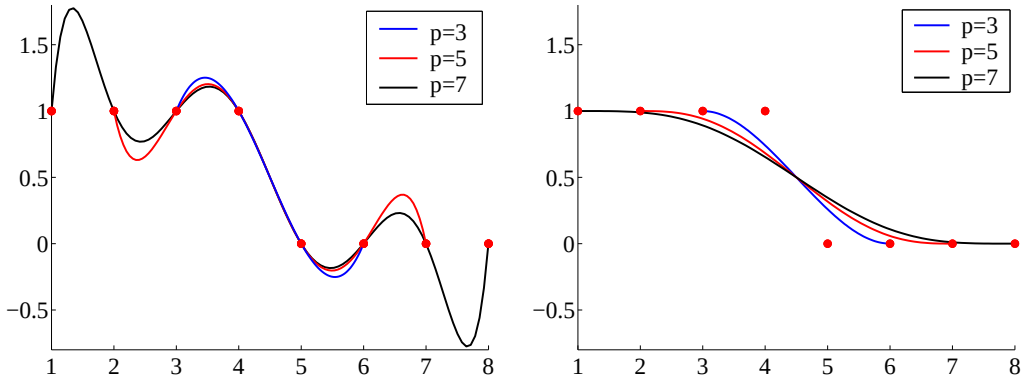


Figure 2.23: (a) Lagrange interpolation oscillates when faced with discontinuous data. (b) NURBS exhibit the variation diminishing property for the same data.

A summary of similar and dissimilar finite element and isogeometric analysis concepts is presented in Table 2.1. A salient feature of isogeometric analysis, shared by meshless methods, is the non-interpolatory nature of the basis. Indeed, a recent NURBS based approach to constructive solid analysis has been described as "meshless" by its authors, see [45]. We disagree with this designation as their method, as well as our current approach, employs a grid defined by the knot vectors throughout the analysis process. What is this grid if not a mesh?

Finite Element Analysis	*Isogeometric Analysis*
Nodal points	Control points
Nodal variables	Control variables
Mesh	Knots
Basis interpolates nodal points and variables	Basis does *not* interpolate control points and variables
Approximate geometry	Exact geometry
Polynomial basis	NURBS basis
Gibbs phenomena	Variation diminishing
Subdomains	Patches
Compact support	
Partition of unity	
Isoparametric concept	
Affine covariance	
Patch tests satisfied	

Table 2.1: Comparison of Finite Element Analysis and Isogeometric Analysis based on NURBS.

2.3 Multiple patches and local refinement

In almost all practical circumstances, it will be necessary to describe domains with multiple NURBS patches. For example, if different material or physical models are to be used in different parts of the domain, it might simplify things to describe these subdomains by different patches. Also, if different subdomains are to be assembled in parallel on a multiple processor machine, it is convenient from the point of view of data structures to not have a single patch split between different processors. Most common is the case where the domain simply differs topologically from a cube. The tensor product structure of the parameter space of a patch makes it poorly suited for representing complex, multiply connected domains. Such geometries can frequently be handled quite simply by using multiple patches (see, *e.g.*, Figure 2.24).

Even in cases where a cube can be mapped into the desired object, doing so might introduce such extreme mesh distortion and widely varying Jacobians within elements that analysis will be adversely affected. Figure 2.25b (from [38]) shows the amount of mesh distortion needed to represent the "stiffened shell" of Figure 2.25a with a single NURBS patch. A mesh using multiple patches, shown in Figure 2.25c, exhibits far less distortion and yields a much more "natural" mesh.

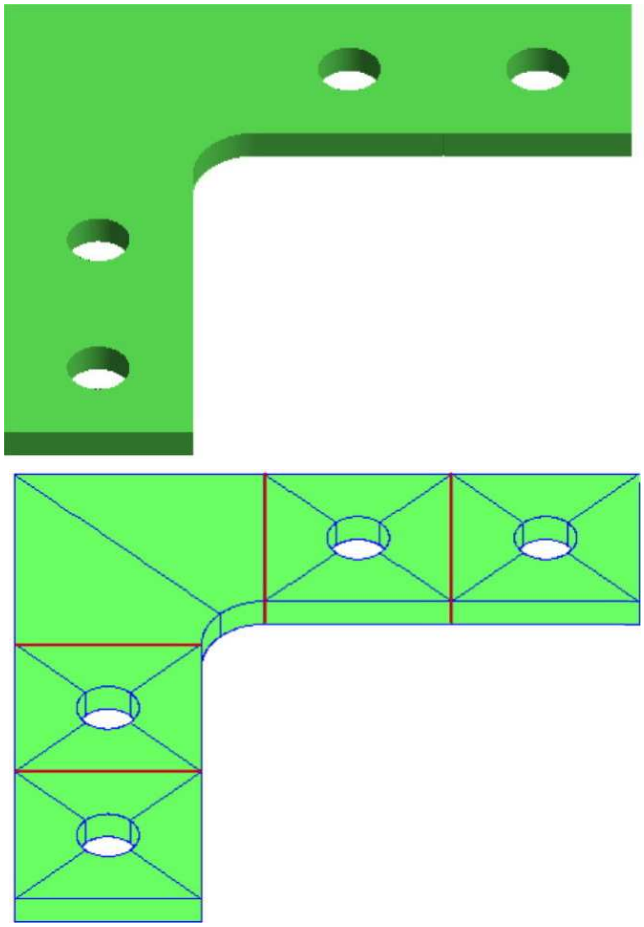
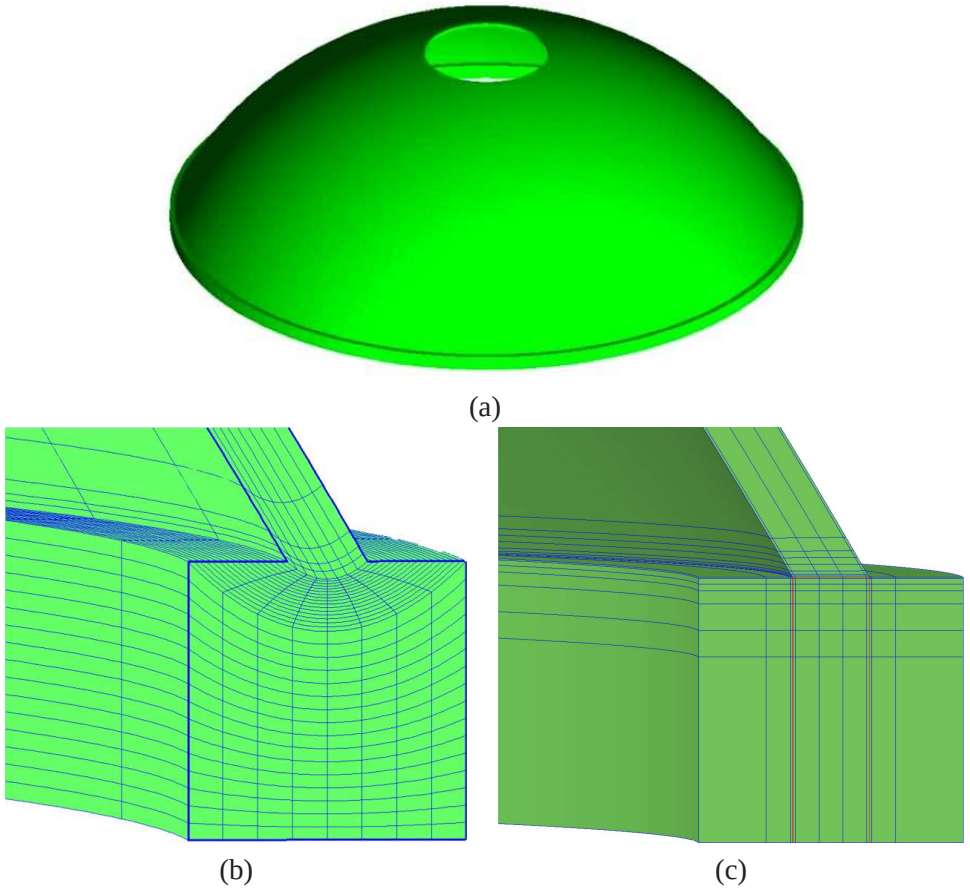
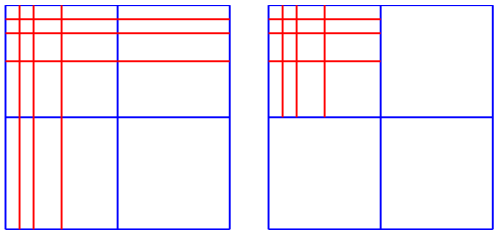


Figure 2.24: The bracket on the top is exactly and concisely represented by five simple NURBS patches (patch boundaries are shown in red, element boundaries in blue). The patches match geometrically and parametrically on the internal faces where they meet.



(a) (b) (c)

Figure 2.25: Multiple patches usually produce better quality meshes. (a) The stiffened shell of [38] can be modeled using a single NURBS patch. (b) Such a mapping produces severe mesh distortion that is unavoidable when using a single patch. (c) Allowing the shell and the stiffener to be modeled by different patches creates a much more natural mesh. Patch boundaries shown in red.



(a) (b)

Figure 2.26: (a) Global refinement employing the continuous Galerkin method. (b) Local refinement employing the discontinuous Galerkin method or constraint equations at the patch level. With constraint equations, at least C^0-continuity can be attained across patches, and higher-order continuity can be achieved in certain cases if desired.

Another reason for using multiple patches is that it makes local refinement possible. The situation is represented in Figure 2.26. Even with multiple patches, if we want the control points of the two patches on their interface to be in one-to-one correspondence, we need to have matching knot vectors. This means that refinements of one patch must necessarily propagate from that patch to the next. If we are to allow knots to be inserted on one side and not the other (*i.e.*, local refinement), we may proceed as follows.

Consider the two B-spline[3] patches that meet on an interface, as shown in Figure 2.27. On the coarsest mesh, we assume that the control points and knot vectors in the plane of the face are identical on both patches, thus ensuring that the patches match geometrically and parametrically on that shared face. Using superscripts 1 and 2 to identify the patch numbers, a subscript f to denote control points on the face where the patches meet, and a subscript n to denote control points *not* on that face, we may write the control points for Patches 1 and 2 as

$$\mathbf{B}^1 = \begin{pmatrix} \mathbf{B}^1_n \\ \mathbf{B}^1_f \end{pmatrix} \quad \text{and} \quad \mathbf{B}^2 = \begin{pmatrix} \mathbf{B}^2_n \\ \mathbf{B}^2_f \end{pmatrix}, \tag{2.16}$$

respectively, where

$$\mathbf{B}^2_f = \mathbf{B}^1_f. \tag{2.17}$$

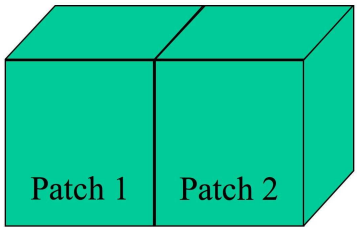


Figure 2.27: The two patches share a common interface. On the coarsest mesh, their control points on that interface are in one-to-one correspondence, trivially enforcing C^0 continuity.

If we now refine the basis of Patch 2 by knot insertion, then we have the following new set of control points for Patch 2:

$$\tilde{\mathbf{B}}^2 = \tilde{\mathbf{T}}\mathbf{B}^2 = \begin{pmatrix} \tilde{\mathbf{T}}_n & 0 \\ 0 & \tilde{\mathbf{T}}_f \end{pmatrix} \begin{pmatrix} \mathbf{B}^2_n \\ \mathbf{B}^2_f \end{pmatrix}, \tag{2.18}$$

where $\tilde{\mathbf{T}}$ is the multi-dimensional generalization of the extension operator defined in (2.7). As

[3]We will discuss the B-spline case here, but it is crucial to note that if we were to use NURBS rather than B-splines, all of the relationships in this section must hold for the *projective* control points and *projective* control variables.

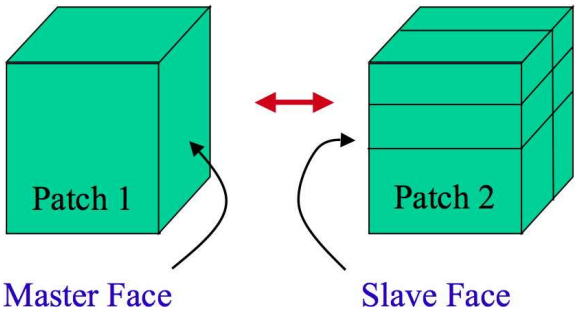


Figure 2.28: As Patch 2 is refined by knot insertion and the one-to-one correspondence of the interface control points is lost. Constraint equations may be employed to ensure that continuity is maintained.

before, it is sparse and its values are entirely defined by the knot vectors and the polynomial order. The block diagonal structure follows from the fact that we are using open knot vectors. When open knot vectors are used, each face of a NURBS solid is influenced only by the control points on that face. Put simply, each face of the NURBS solid is a NURBS surface.

Combining (2.17) and (2.18), we see that C^0-continuity of the geometry is maintained by the relationship

$$\tilde{\mathbf{B}}_f^2 = \tilde{\mathbf{T}}_f \mathbf{B}_f^1. \tag{2.19}$$

Building on the approach of Kagan, Fischer and Bar-Yoseph [43][4], it follows that for our solution space to enforce the same continuity constraints, we need our control variables to obey precisely the same relationship. Let

$$\mathbf{u}^1 = \begin{pmatrix} \mathbf{u}_n^1 \\ \mathbf{u}_f^1 \end{pmatrix} \quad \text{and} \quad \mathbf{u}^2 = \begin{pmatrix} \mathbf{u}_n^2 \\ \mathbf{u}_f^2 \end{pmatrix} \tag{2.20}$$

be the control variables on Patch 1 and the refined Patch 2, respectively. Then C^0-continuity of the solution across the interface between the patches may be maintained by enforcing the constraint

$$\mathbf{u}_f^2 = \tilde{\mathbf{T}}_f \mathbf{u}_f^1. \tag{2.21}$$

From an implementational point of view, the two patches may be assembled locally to create the two local problems

$$\mathbf{K}^1 \mathbf{u}^1 = \mathbf{b}^1 \tag{2.22}$$

and

$$\mathbf{K}^2 \mathbf{u}^2 = \mathbf{b}^2 \tag{2.23}$$

[4] In [43], a similar approach was taken for B-Splines *surfaces*. Here we extend that to NURBS *solids*.

for the control points on either patch. Consistent with the partitioning of the control variables in (2.20), we partition the stiffness matrices as

$$\mathbf{K}^1 = \begin{pmatrix} \mathbf{K}^1_{nn} & \mathbf{K}^1_{nf} \\ \mathbf{K}^1_{fn} & \mathbf{K}^1_{ff} \end{pmatrix} \quad \text{and} \quad \mathbf{K}^2 = \begin{pmatrix} \mathbf{K}^2_{nn} & \mathbf{K}^2_{nf} \\ \mathbf{K}^2_{fn} & \mathbf{K}^2_{ff} \end{pmatrix}. \tag{2.24}$$

Before solving, we must assemble problems (2.22) and (2.23) into one global problem accounting for the behavior of both patches, as well as their interaction. We should have three coupled blocks of equations: one corresponding to weighting functions with support in Patch 1 that vanish on the face shared by the two patches, one corresponding to weighting functions with support on either or both patches that do *not* vanish on the shared face, and one corresponding to weighting functions with support on Patch 2 that vanish on the shared face. We begin by expanding (2.22) using the partitioning of (2.24) to get

$$\mathbf{K}^1_{nn}\mathbf{u}^1_n + \mathbf{K}^1_{nf}\mathbf{u}^1_f = \mathbf{b}^1_n \tag{2.25}$$

and

$$\mathbf{K}^1_{fn}\mathbf{u}^1_n + \mathbf{K}^1_{ff}\mathbf{u}^1_f = \mathbf{b}^1_f. \tag{2.26}$$

Inserting (2.21) into (2.23) and expanding yields

$$\mathbf{K}^2_{nn}\mathbf{u}^2_n + \mathbf{K}^2_{nf}\tilde{\mathbf{T}}_f\mathbf{u}^1_f = \mathbf{b}^2_n \tag{2.27}$$

and

$$\mathbf{K}^2_{fn}\mathbf{u}^2_n + \mathbf{K}^2_{ff}\tilde{\mathbf{T}}_f\mathbf{u}^1_f = \mathbf{b}^2_f. \tag{2.28}$$

Note that (2.25) is the block of equations corresponding to weighting functions in Patch 1 that vanish on the shared face. Similarly, (2.27) is the block of equations corresponding to weighting functions in Patch 2 that vanish on the shared face. Now (2.26) and (2.28) both correspond to weighting functions with support on the shared face and as such we would like to add them together to get a final expression for that block. Unfortunately, they contain different numbers of equations. This is because we assembled the two patches independently. We correctly generated the equations in (2.26) by testing against functions in the "master" weighting space associated with Patch 1, but we generated the equations in (2.28) by testing against all of the functions in the larger "slave" weighting space on Patch 2 without regard for the constraint. Just as the basis functions of the slave solution space on Patch 2 corresponding to the shared face are restricted to act only in the linear combinations defined by $\tilde{\mathbf{T}}_f$ that result in functions existing in the master solution space, so too must the functions in the slave weighting space act only in such linear combinations as replicate

functions in the master weighting space. This constraint may be enforced by now premultiplying (2.28) by $\tilde{\mathbf{T}}_f^{\mathrm{T}}$, thus constraining the weighting functions and reducing the number of equations to match that of (2.26):

$$\tilde{\mathbf{T}}_f^{\mathrm{T}}\mathbf{K}_{fn}^2\mathbf{u}_n^2 + \tilde{\mathbf{T}}_f^{\mathrm{T}}\mathbf{K}_{ff}^2\tilde{\mathbf{T}}_f\mathbf{u}_f^1 = \tilde{\mathbf{T}}_f^{\mathrm{T}}\mathbf{b}_f^2. \tag{2.29}$$

We may now express the global system comprised of (2.25), (2.27), and $\big((2.26)+(2.29)\big)$ as

$$\mathbf{K}\mathbf{u} = \mathbf{b}, \tag{2.30}$$

where

$$\mathbf{K} = \begin{pmatrix} \mathbf{K}_{nn}^1 & \mathbf{K}_{nf}^1 & 0 \\ \mathbf{K}_{fn}^1 & (\mathbf{K}_{ff}^1 + \tilde{\mathbf{T}}_f^{\mathrm{T}}\mathbf{K}_{ff}^2\tilde{\mathbf{T}}_f) & \tilde{\mathbf{T}}_f^{\mathrm{T}}\mathbf{K}_{fn}^2 \\ 0 & \mathbf{K}_{nf}^2\tilde{\mathbf{T}}_f & \mathbf{K}_{nn}^2 \end{pmatrix}, \tag{2.31}$$

$$\mathbf{u} = \begin{pmatrix} \mathbf{u}_n^1 \\ \mathbf{u}_f^1 \\ \mathbf{u}_n^2 \end{pmatrix}, \tag{2.32}$$

and

$$\mathbf{b} = \begin{pmatrix} \mathbf{b}_n^1 \\ \mathbf{b}_f^1 + \tilde{\mathbf{T}}_f^{\mathrm{T}}\mathbf{b}_f^2 \\ \mathbf{b}_n^2 \end{pmatrix}. \tag{2.33}$$

We may recover $\mathbf{u}_f^2$ via (2.21) after solving (2.30).

This approach ensures C^0 continuity in the solution across the patch boundary when one patch is a knot refined version of the other patch on their common interface. Higher continuity has also been implemented by applying similar constraint equations in the normal direction. As long as the geometries are compatible, the patch boundary may be seen as the result of inserting a knot into some "metapatch" $p+1$ times. It should be noted that these are strong, exact constraints, not approximations. An approach that would allow for weak enforcement of continuity, as well as allowing for local order elevation is to use discontinuous Galerkin techniques at the patch level. That is, weakly enforce continuity of appropriate fluxes across patch boundaries while strongly enforcing them across element boundaries within the patch.

Remark 2.3.1. *It is important to note that these operations could also be applied over the entire domain rather than just for the interface between patches. This could be used in a multigrid scheme where the grid transfer operator would be* $\tilde{\mathbf{T}}$*, which could potentially be very efficient as* $\tilde{\mathbf{T}}$ *is uniquely defined by the knot vectors, and thus its construction is very inexpensive.*

Chapter 3

Selected Numerical Examples in Isogeometric Analysis

The first NURBS based isogeometric analysis solver was for linear elasticity. It was used to investigate a number of structures modeled by single patches. Geometrically exact solid models were made, even for structures which might normally be treated by shell models. Good results were obtained and optimal convergence rates observed[1]. Soon afterward, the advection-diffusion problem was examined. Excellent results were seen for a *k*-refinement strategy in the presence of sharp internal and boundary layers, a very surprising result likely due to the treatment of the boundary conditions. Eventually the code was expanded to tackle domains defined by multiple patches. Local refinement using constraint equations of the type described in the previous section was implemented. Examination of structural vibrations showed the most convincing evidence yet of the power of a *k*-refinement strategy, and the modeling of the NASA Aluminum Testbed Cylinder demonstrated the feasibility of creating exact geometrical models of real structures of engineering interest. This chapter highlights some selected results from throughout the development of isogeometric analysis. These particular examples were specifically chosen to emphasize the major features of the method: geometric flexibility, functions of high continuity, and local refinement. For a more in depth discussion of these problems, as well as many more, see [38], [19], [6], [59] and [18].

3.1 Linear elasticity

NURBS based isogeometric analysis is naturally suited to the study of linear elasticity. The problems are largely governed by geometry (recall Figure 1.3) and so the concept of an exact encapsu-

[1] A convergence proof now exists explaining the observed convergence rates. See Bazilevs, Beiro da Veiga, Cottrell, Hughes and Sangalli [6]

lation of that geometry at every level of mesh refinement is certainly attractive. Interesting results are obtained for single-patch, linear elasticity by treating thin shelled structures as solids. We are pleased by the robustness of the NURBS approach in this context. For example, throughout our work we looked at both direct and iterative linear algebraic solvers, but the original code we developed was restricted to a single processor. Many of the finer-mesh cases were memory bound and could not be solved with the direct solver because they exceeded available resources. The aspect ratios of elements in some cases were quite large and the number of equations approached a quarter of a million. Nevertheless, in all cases the iterative procedure converged without difficulty. Though we did not look at the corresponding finite element cases, we would be surprised if they converged as reliably. Consequently, we suspect that the NURBS cases may be better conditioned (perhaps due to the larger support of the functions) than the corresponding finite element cases. Though we have only indirect evidence to support such a claim, it at least warrants investigation.

3.1.1 Thin cylindrical shell with fixed ends subjected to constant internal pressure

This problem, featured in [38], was one of the first demonstrations of the ability of a NURBS based code with smooth basis functions (*i.e.*, high-order achieved via *k*-refinement) to accurately capture a boundary layer. The problem setup and a radial displacement profile are shown in Figure 3.1. This boundary layer, due to the fixed ends of the cylinder, is difficult to accurately capture with low order finite elements. The exact shell theory solution is given in Timoshenko and Woinowsky-Krieger [56], pp. 476-477, for plane stress and is provided as a reference below:

$$u(x) = -\frac{PR^2}{Et}(1 - C_1 \sin\beta x \sinh\beta x - C_2 \cos\beta x \cosh\beta x) \tag{3.1}$$

$$x \in (-L/2, L/2),$$

$$C_1 = \frac{\sin\alpha\cosh\alpha - \cos\alpha\sinh\alpha}{\sinh\alpha\cosh\alpha + \sin\alpha\cos\alpha}, \tag{3.2}$$

$$C_2 = \frac{\cos\alpha\sinh\alpha + \sin\alpha\cosh\alpha}{\sinh\alpha\cosh\alpha + \sin\alpha\cos\alpha}, \tag{3.3}$$

$$\beta = (\frac{Et}{4R^2D})^{1/4}, \quad \alpha = \frac{\beta L}{2}, \quad D = \frac{Et^3}{12(1-\nu^3)} \tag{3.4}$$

The Young's modulus and Poisson's ratio in this solution need to be replaced by $\frac{E}{1-\nu^2}$ and $\frac{\nu}{1-\nu}$, respectively, to account for the fixed-end conditions assumed here. The geometry of the shell is shown in Figure 3.2. Note that the radius to thickness ratio is 100. The meshes are depicted in Figures 3.3 and 3.4. The first four surface meshes are shown in Figure 3.3. Note the added refinement in the region of the boundary layer. We wish to emphasize that, despite the shell being very thin, we are modeling it with solid elements. (For an excellent comprehensive review of approaches to shell modeling, see Bischoff *et al.* [9].)

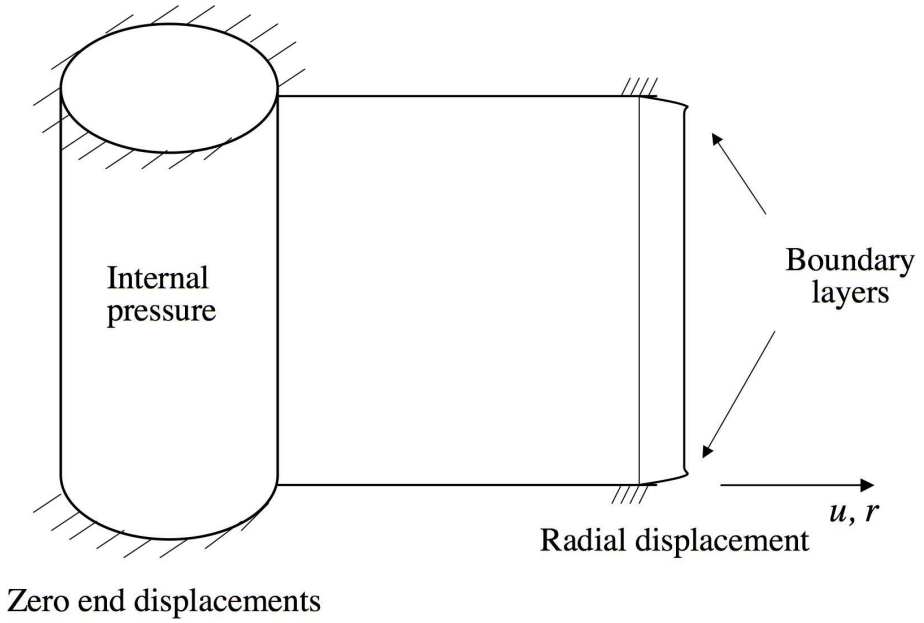


Figure 3.1: Thin cylindrical shell. Problem statement and displacement profile.

The convergence of the radial displacement profile is shown on Figure 3.5. Mesh 1 is too coarse to represent both of the boundary layers as well as the plateau between them. Mesh 3 picks up the plateau but the boundary layers are still not accurately captured. The Mesh 5 solution is indistinguishable from the exact shell theory solution. In the detail on the right, the exact shell solution and Mesh 5 solution are seen to overlap in the boundary layer region.

3.1.2 Hemispherical shell with a stiffener

The hemispherical shell with a stiffener problem (see Figure 3.6) was modeled with a single NURBS patch in [38]. As was shown in Section 2.3, use of a single patch leads to undesirable distortion of the elements. While it speaks well of the overall robustness of the method that accurate results were still obtained, efficiency clearly suffered. The *p*-method used in that convergence study was not competitive on a per degree-of-freedom basis with the original results of Rank *et al.* [47], who used a trunk space p-refinement strategy. Such an approach does not use the full tensor product space of basis functions, but the much smaller *trunk* space, just large enough to ensure the optimal convergence rate at a given polynomial order (see Szabó, Düster and Rank [55] for a discussion of the trunk space and the *p*-method in general). As NURBS necessarily have an underlying tensor product structure, at least on patches, an analogous isogeometric analysis approach exploiting the trunk space has not been attempted thus far.

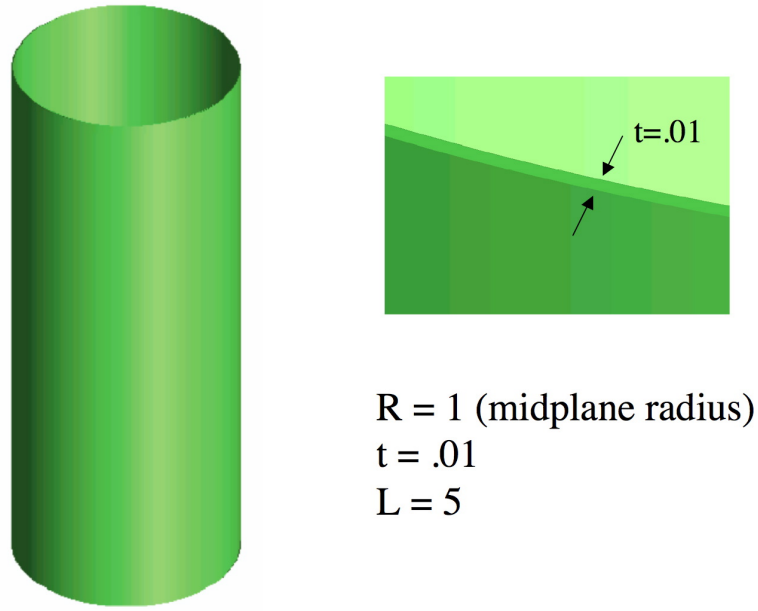


Figure 3.2: Thin cylindrical shell geometry.

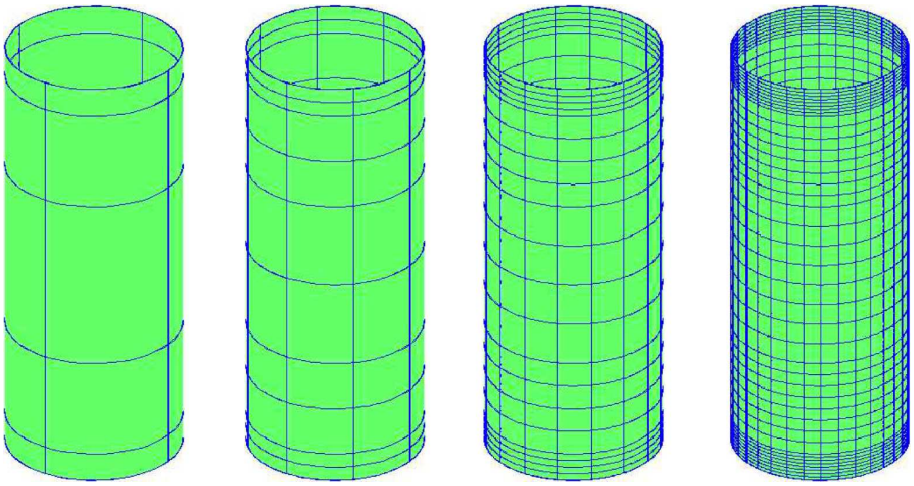
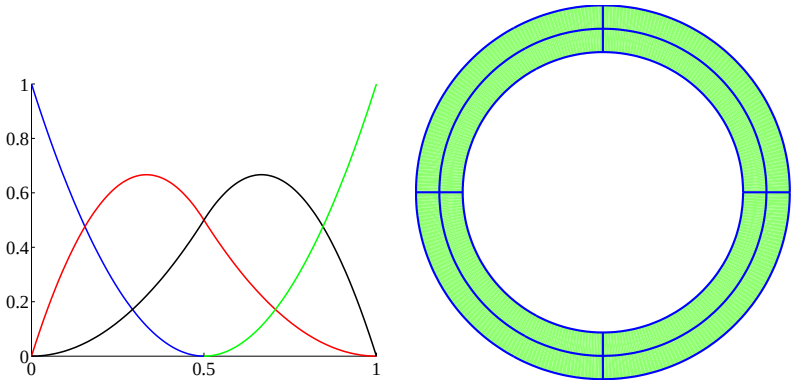
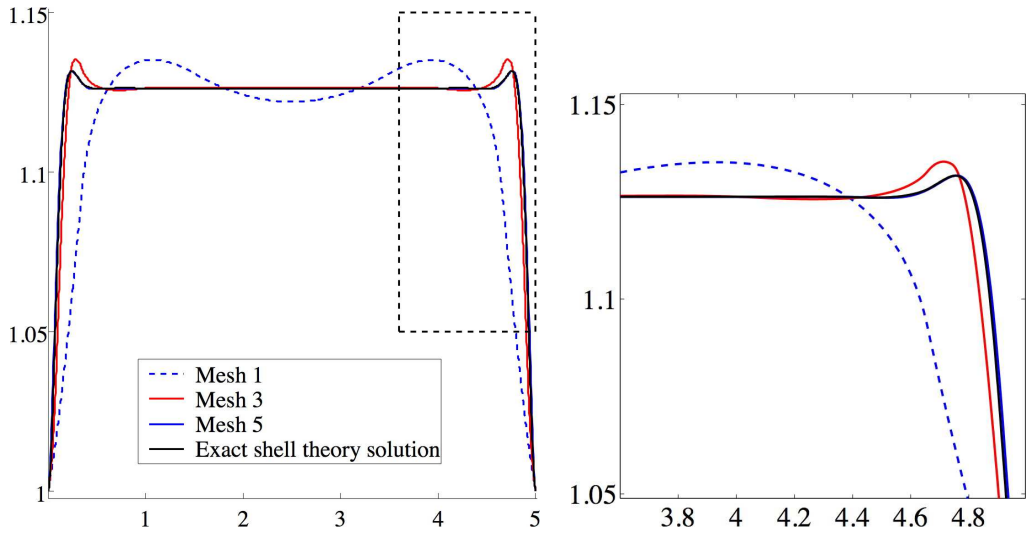


Figure 3.3: Thin cylindrical shell surface meshes. Meshes 1-4.



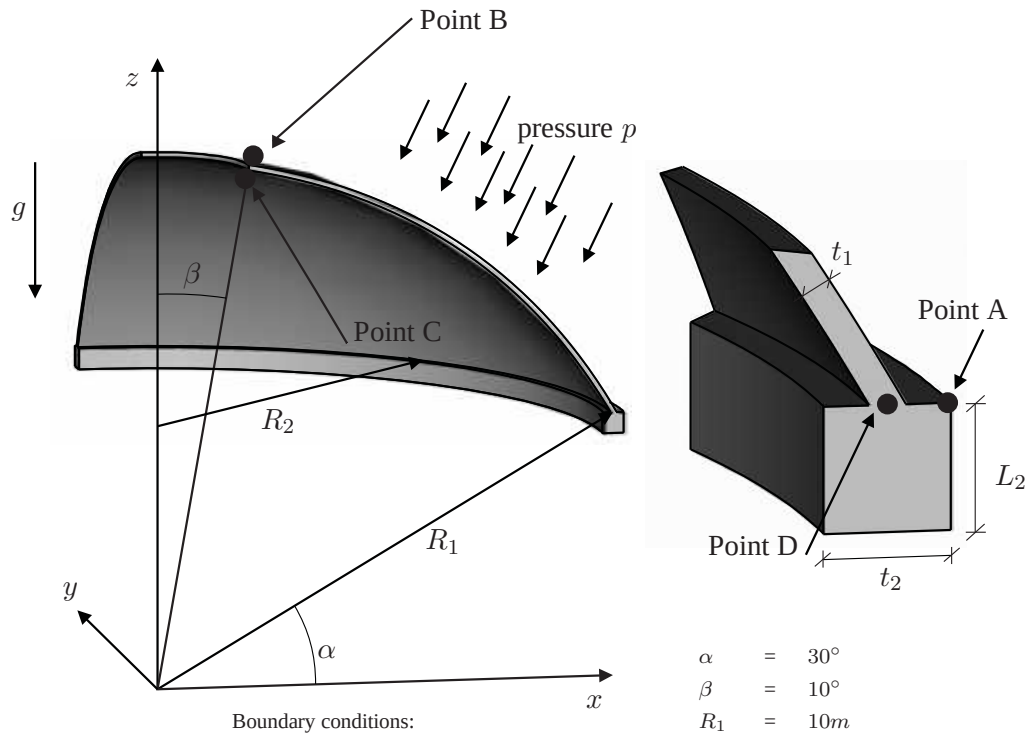
(a) (b)

Figure 3.4: Thin cylindrical shell. (a) Quadratic basis functions through the thickness. (b) End view of the coarse mesh.



(a) Global radial displacement (b) Detail of the boundary layer

Figure 3.5: Thin cylindrical shell. Convergence of radial displacement to exact shell theory solution. Mesh 5 solution indistinguishable from exact. NURBS are capable of accurately resolving shell boundary layers.



Boundary conditions:

At bottom surface of stiffener:

$u_z = 0$

Symmetry at x-z-plane:

$u_y = 0$

Symmetry at y-z-plane:

$u_x = 0$

α	=	$30°$
β	=	$10°$
R_1	=	$10m$
R_2	=	$5\sqrt{3}m$
t_1	=	$0.1m$
t_2	=	$0.4m$
L_2	=	$0.4m$
E	=	$6.825 \cdot 10^7 \frac{kN}{m^2}$
ν	=	0.3
ρ	=	$500 \frac{kg}{m^3}$
g	=	$10.0 \frac{m}{s^2}$
p	=	$100.0 \frac{kN}{m^2}$

Figure 3.6: Hemispherical shell with stiffener. Problem description from Rank *et al.* [47].

Despite the tensor product structure of NURBS, k-refinement presents the possibility of improved efficiency. In fact, *k*-refinement, in conjunction with the use of multiple patches to create better quality meshes, and the use of local refinement to avoid placing functions in regions where they are not needed, enables the NURBS based approach to show an accuracy per degree-of-freedom comparable to the results presented by Rank *et al.* in [47]; see Figures 3.9-3.12.

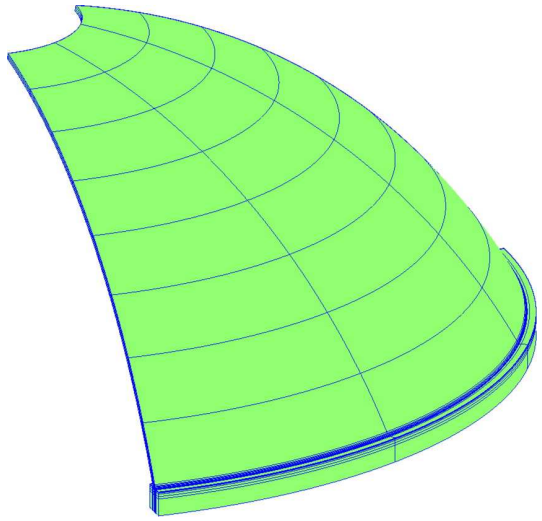
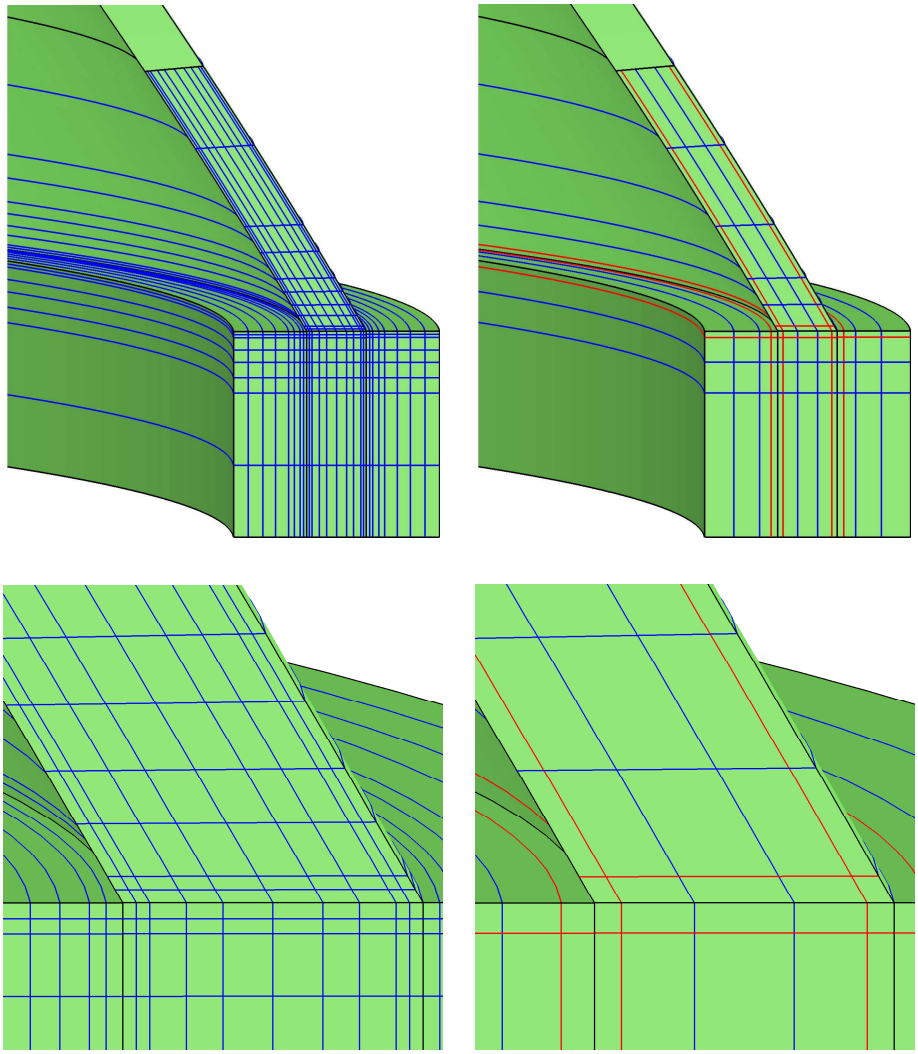


Figure 3.7: Hemispherical shell with stiffener. The coarse mesh may be refined in multiple ways.

In studying the stiffened shell problem with a *k*-refinement approach, certain previously unobserved features begin to emerge. As above, if a given mesh of higher-order and high continuity does not achieve the level of accuracy desired, one can add more degrees-of-freedom by inserting a knot in one of the parametric directions. The number of new degrees-of-freedom is *exactly* the same regardless of whether a new knot value is inserted (creating new elements by splitting existing ones), or whether an existing knot value is repeated (creating no new elements, but decreasing the continuity of the basis across the corresponding element boundaries). While a rigorous analysis of the two approaches has not yet been performed, in the present results it seems clear that in regions where the solution is very smooth (such as in the shell, a reasonable distance away from the stiffener), inserting a new knot, and thus more functions that maintain high continuity, was the more beneficial refinement; see Cottrell *et al.* [19] for a more thorough discussion. In the vicinity of a singularity (such as near the reentrant corner where the shell meets the stiffener and the stress is singular), it is more beneficial to repeat an existing knot value, decreasing the continuity of the basis and simultaneously decreasing the support of the basis functions in the physical space. Both of these effects help localize the singularity and prevent it from polluting the results elsewhere in



(a) Local k-refinement (b) Local k^*-refinement

Figure 3.8: Hemispherical shell with stiffener. (a) A *k*-refinement approach with C^{p-1} continuity across element boundaries. Many small elements are used to get a well-resolved solution. (b) Functions are C^0 across the element boundaries in red, C^{p-1} elsewhere. Fewer elements are needed than in (a). In both cases, the basis is C^0 across patch boundaries, shown in black, and local refinement is implemented at the patch level.

the domain[2].

The meshes for the multiple-patch treatment of the stiffened shell are shown in Figures 3.7 and 3.8. The locally refined, *k*-method meshes are seen in Figure 3.8a. In Figure 3.8b, we see the case where fewer elements are used. A *k*-type refinement is used everywhere except at the knot lines marked in red. The multiplicities of these knots were increased with the polynomial order such that the basis remained C^0 across them. The results for this mesh are labeled "Local *k**-ref" to indicate that the *k*-refinement paradigm was altered near the singularity. The displacements are plotted versus the number of degrees-of-freedom in Figures 3.9-3.12. The calculated von Mises stresses are plotted versus the number of degrees-of-freedom in Figures 3.13-3.16. The trunk space *p*-method results from Rank *et al.* [47] are plotted for comparison. For displacements, the single patch results from [38] are plotted as well.

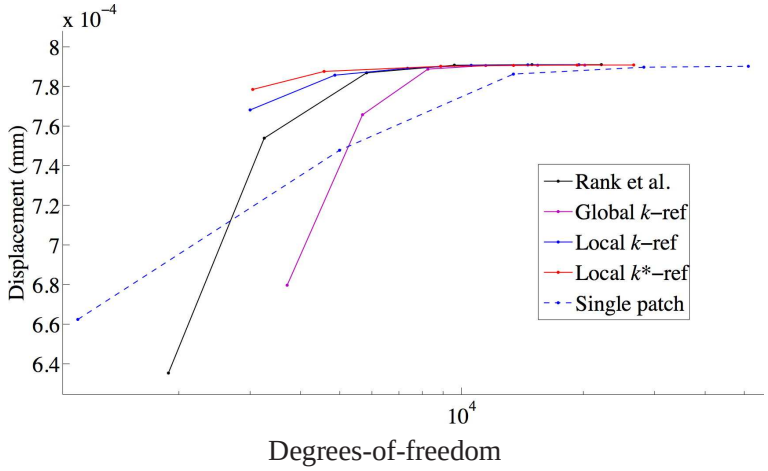


Figure 3.9: Hemispherical shell with stiffener. The displacement at point A is plotted versus the total number of degrees-of-freedom.

[2]This is reminiscent of the heuristic notion that an *hp*-method should use large elements with higher-order in smooth regions and small elements of lower-order near singularities. Coupling this with control over the continuity across elements opens the door to the possibility of an *hpk*-method.

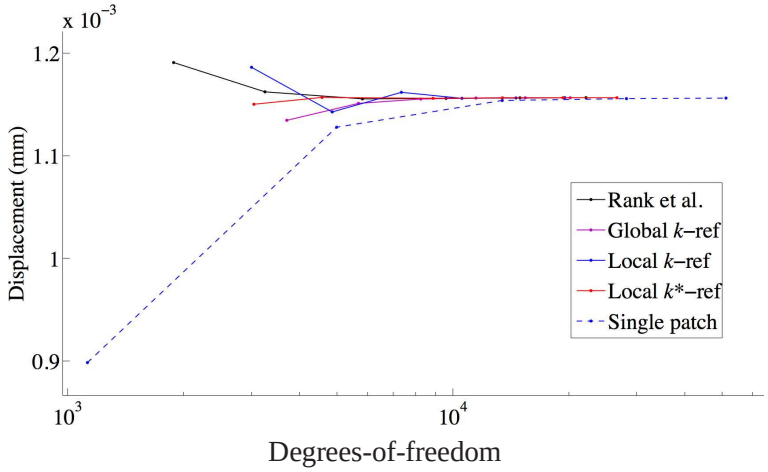


Figure 3.10: Hemispherical shell with stiffener. The displacement at point B is plotted versus the total number of degrees-of-freedom.

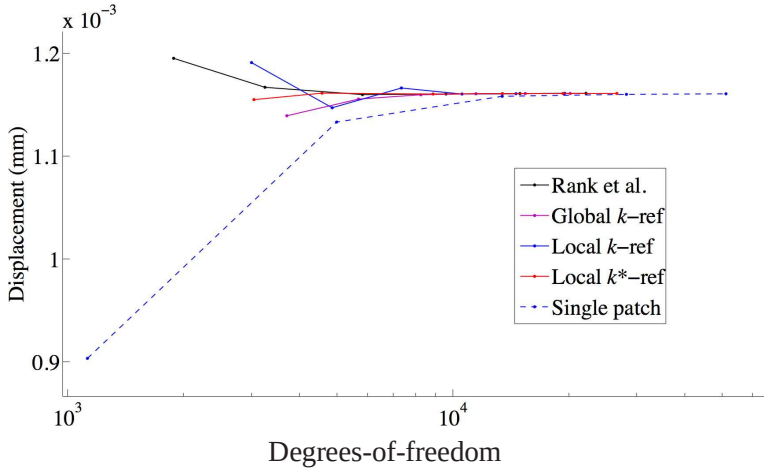


Figure 3.11: Hemispherical shell with stiffener. The displacement at point C is plotted versus the total number of degrees-of-freedom.

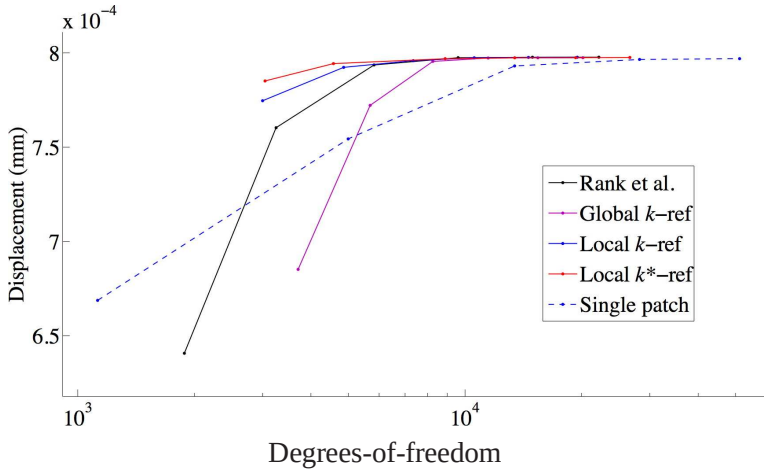


Figure 3.12: Hemispherical shell with stiffener. The displacement at point D is plotted versus the total number of degrees-of-freedom.

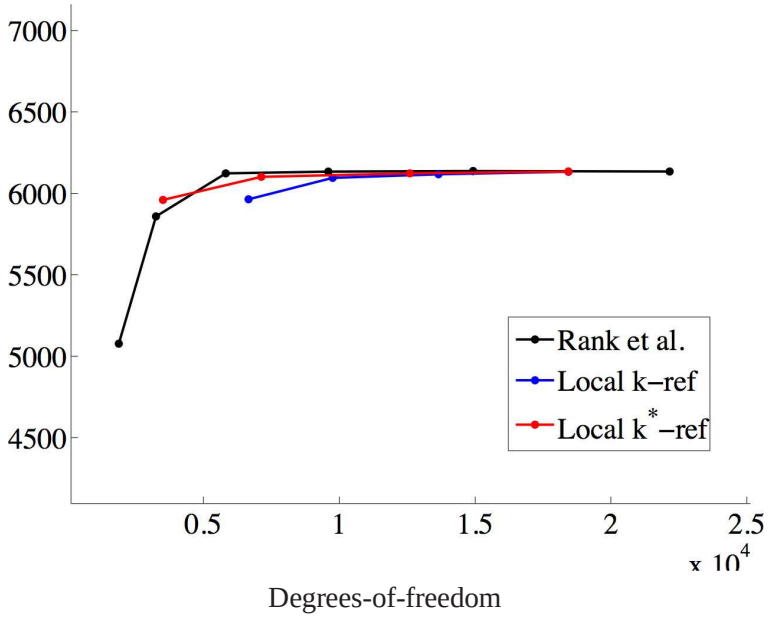


Figure 3.13: Hemispherical shell with stiffener. The von Mises stress at point A is plotted versus the total number of degrees-of-freedom.

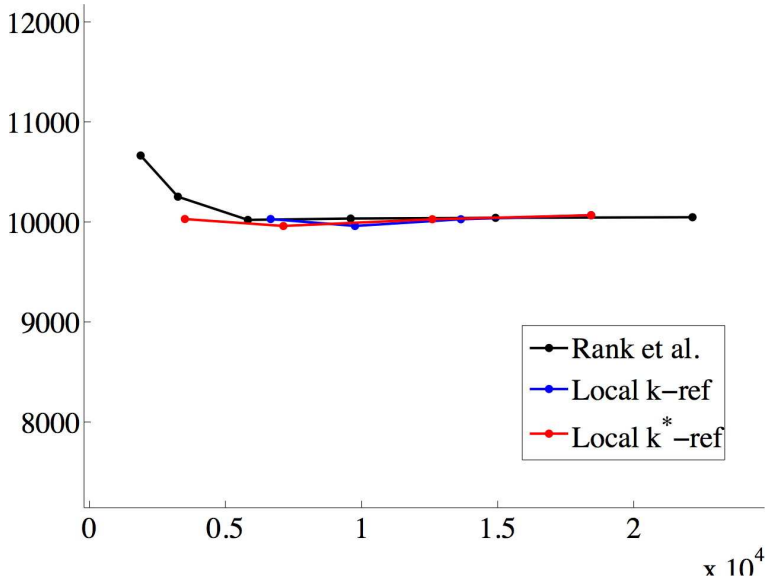


Figure 3.14: Hemispherical shell with stiffener. The von Mises stress at point B is plotted versus the total number of degrees-of-freedom.

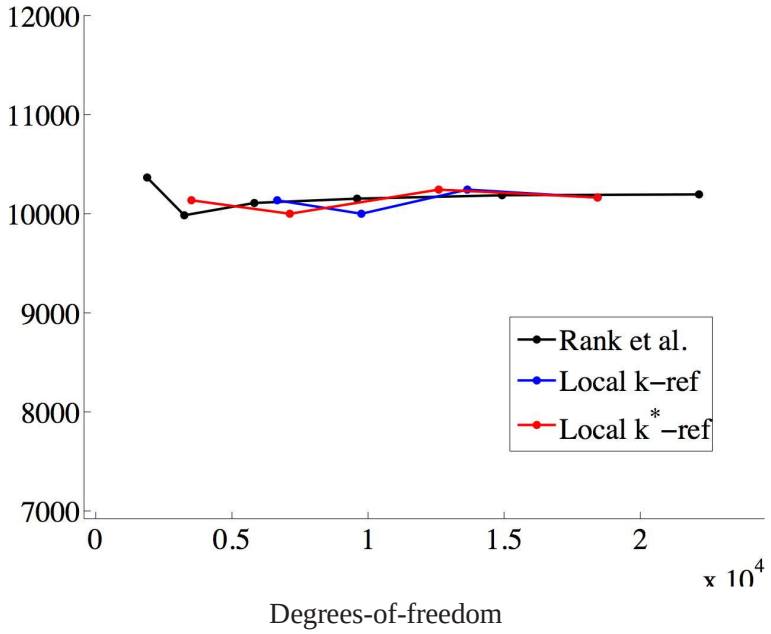


Figure 3.15: Hemispherical shell with stiffener. The von Mises stress at point C is plotted versus the total number of degrees-of-freedom.

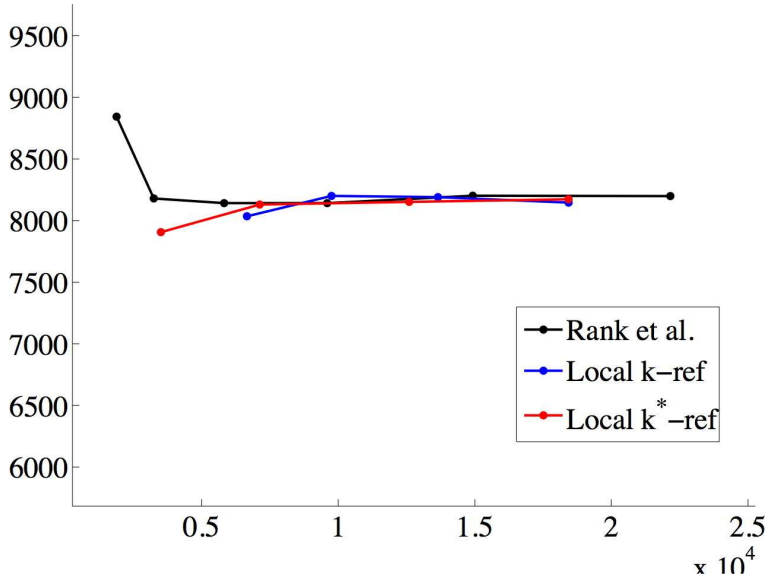


Figure 3.16: Hemispherical shell with stiffener. The von Mises stress at point D is plotted versus the total number of degrees-of-freedom.

3.2 Advection-diffusion

In this section we investigate the ability of the isogeometric approach, in conjunction with SUPG, to solve a challenging test case for the advection-diffusion equation. Isogeometric analysis is fundamentally a higher-order approach and one might not expect good behavior in situations with unresolved interior and boundary layers. In fact, the Gibbs phenomena noted for polynomial-based finite element methods tends to become more pronounced as polynomial order is increased. This is the reason that most practical fluids formulations employ lower-order, typically constant and linear, interpolation of flow variables. However, the variation diminishing property of the Dirichlet boundary condition specification, plus the notion of k-refinement, leads to some remarkable results in the case of NURBS, first shown in [38].

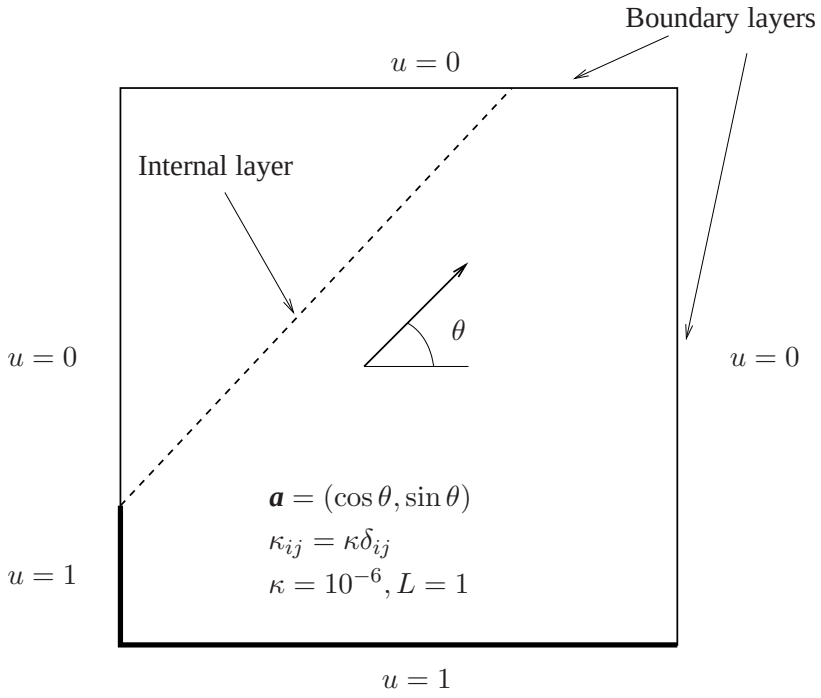


Figure 3.17: Advection skew to mesh. Problem description and data.

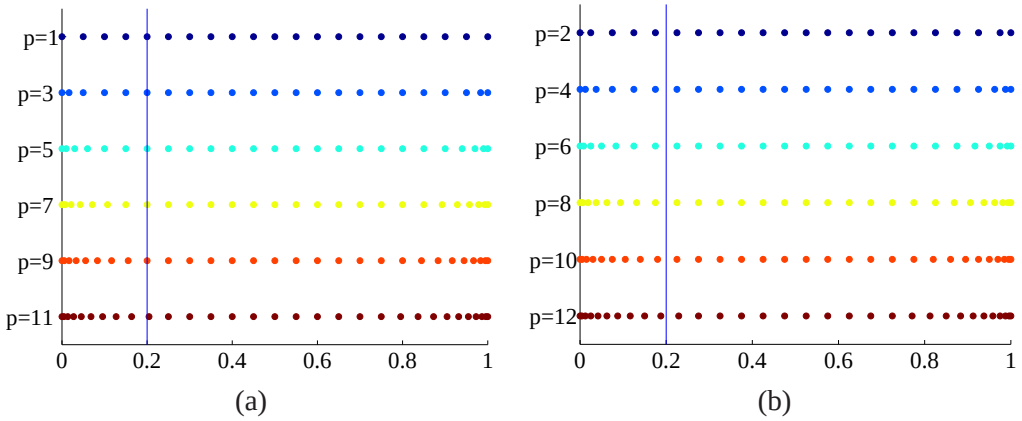


Figure 3.18: The y-coordinate of the control points along the left edge of the domain.(a) Odd polynomial orders. (b) Even polynomial orders.

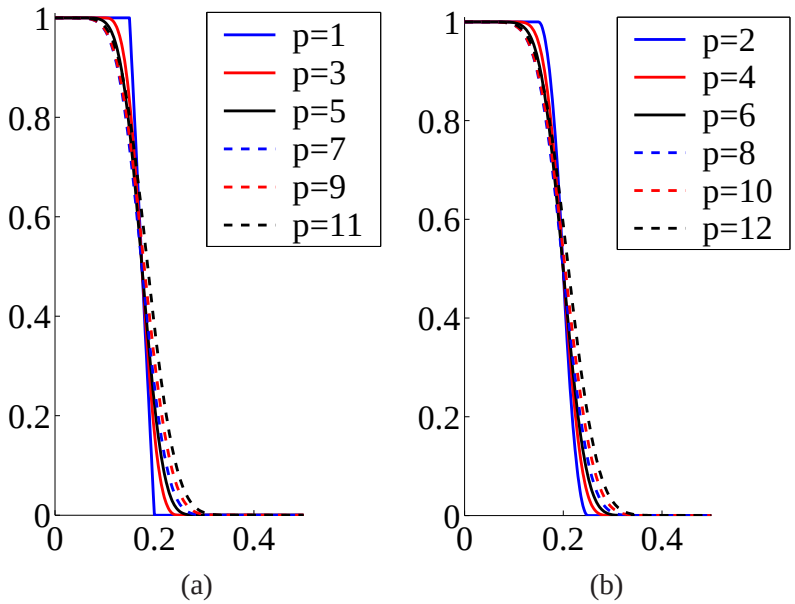


Figure 3.19: Dirichlet boundary conditions along the left edge of the domain.(a) Odd polynomial orders. (b) Even polynomial orders.

The problem setup is described in Figure 3.17. Here, a is the advective velocity magnitude, κ is the diffusivity, and L is the side length of the domain. The Peclet number, $Pe = aL/\kappa = 10^6$. When this number is greater than one, advection dominates and diffusion is only important in very small layers. In the present case, diffusion is important in a region of thickness $\mathrm{O}(Pe^{-1}\ln Pe)$ in the outflow boundary layers and $\mathrm{O}(Pe^{-1/2}\ln Pe)$ in the internal layer (see Wahlbin [57], pp. 468). In all calculations the mesh is uniform, consisting of a 20×20 grid of square elements, with element side length $h = 1/20 = 0.05$. Refinement is performed by the k-method, and solutions from $p = 1$ to $p = 12$ are calculated. In all cases, the standard SUPG formulation is used with $\tau = h_a/2a$, where h_a is the element length in the direction of the flow velocity which, in the present case, is simply, $h_a = h/\max\{\cos\theta, \sin\theta\}$.

The boundary condition is set by specifying the control variables. On the top and right edges of the domain, all control variables are set to 0 and the boundary condition is exactly satisfied along these edges. On the bottom, the control variable corresponding to the lower right-hand corner is set to 0 and the remainder are set to 1. The result is that the boundary value is identically 1 up to the last element in which it smoothly decreases to 0 at the corner. The left-hand-side boundary is more interesting. If we think of our control variables as control points in $\mathbb{R}^3$ defining the surface plot of the solution, where the x and y coordinates have been fixed by the two-dimensional geometrical mapping and are no longer to be chosen by the user, then what we have done along the left side of our domain is to set the z-component (our actual control variable) equal to 1 for each control point that falls in the interval $[0, 0.2)$, and equal to 0 if it falls in $[0.2, 1]$. The locations of the control points are shown in Figure 3.18. Note the clustering of points near the open knots. The resulting boundary conditions are shown in Figure 3.19. For $p = 1$, the boundary condition is interpolated, whereas for $p > 1$ it is fit to the control variables in monotone fashion as the variation diminishing property of B-splines prevents the curve from over- and under-shooting. We wish to emphasize that *k*-refinement produces non-nested solution spaces, which prevents us from having exactly the same boundary condition at each stage of the refinement process. As a result of this technique, the discontinuity "smears" about the location 0.2. For odd polynomial orders with $p \leq 9$, a control point falls directly on 0.2, whereas for even polynomial orders one does not (see Figure 3.18). When p is larger than 9, the "fringing" due the open knot vectors disrupts this pattern. Clearly, a better scheme for setting the boundary control variables is required. One possibility is to obtain the desired boundary condition by setting up a projection problem over the boundary, for which a suitable norm needs to be chosen. Nevertheless, the main point of the present study is to assess the ability of NURBS (in this case, B-splines because of the simplicity of the domain) to deal with unresolved boundary and interior layers. We considered the case in which $\theta = 45^\circ$ and in which $\theta = \tan^{-1}(2)$, both for $p = 1$ to $p = 12$. Selected results for $\theta = \tan^{-1}(2)$ are shown in Figure 3.20 (see [38] for all of the cases examined). Two views are presented for each p, one in which the plotting routine

sampled the solution with a 100×100 grid of uniformly distributed points and one in which it is sampled with a 21 $\times$ 21 uniform grid. In the former case the plot is Phong shaded and in the latter, it is represented by bilinear interpolation on each element and the element edges are drawn. The philosophy behind the dual views is that the 100 $\times$ 100 grid plots are a more faithful rendering of the higher-order cases, whereas the 21 $\times$ 21 point piecewise bilinear interpolates are the type of plots that have appeared in numerous research articles over the years and these may be more easily visually compared with results in the literature. For $p = 1$, the overshoot in the outflow boundary layer is approximately 45%. As one examines the results, it is clear that they improve as p increases and are converging toward monotone results with quite sharp layers. One would perhaps expect that oscillations would increase with increasing p but this is not the case. This is certainly due in part to the smearing of the boundary condition but we suspect that the high continuity of the basis obtained through *k*-refinement plays a part in this as well, particularly in the outflow layer. Further studies need to be undertaken to clarify these issues.

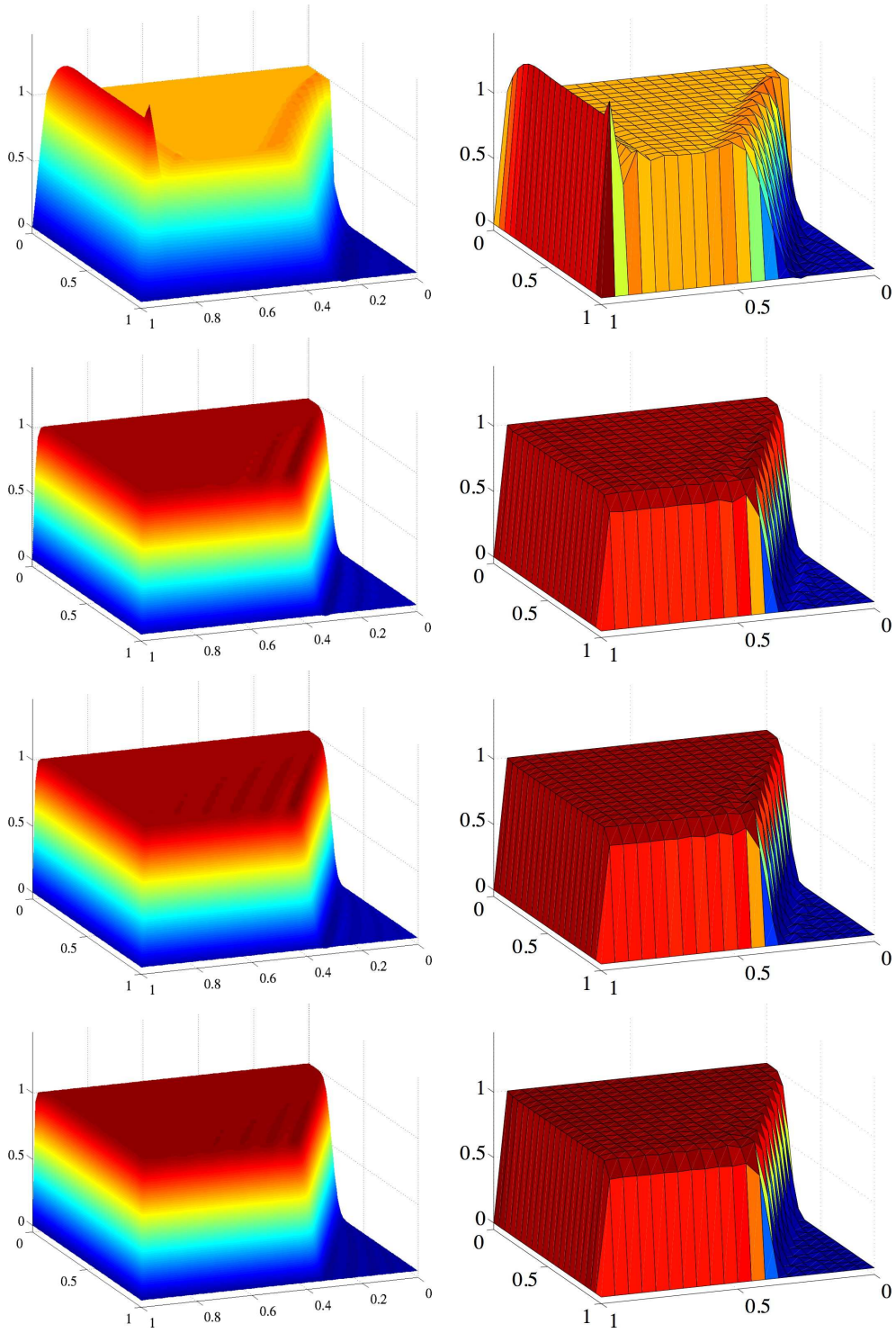


Figure 3.20: Advection skew to the mesh, $\theta \approx 63.4°$. Top to bottom: results for $p = 1$, $p = 5$, $p = 8$, and $p = 12$. Left: plot with 100 $\times$ 100 points, Phong shaded. Right: plot with 21 $\times$ 21 points.

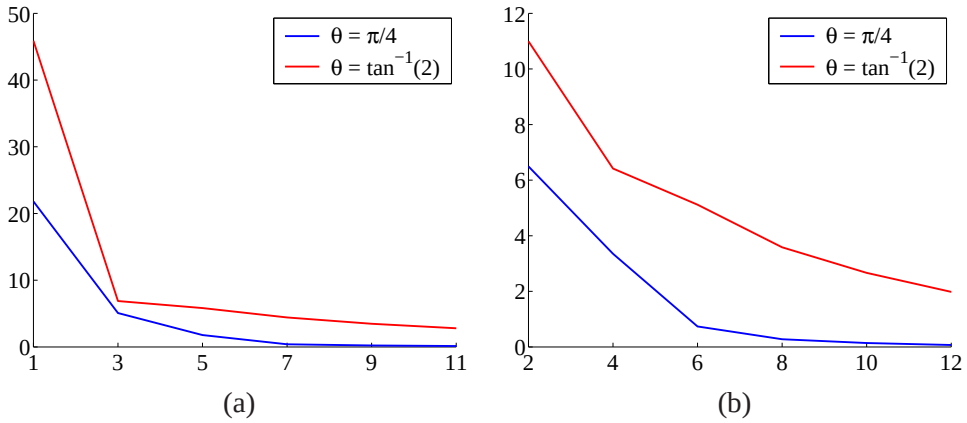


Figure 3.21: Maximum solution over-shoot versus polynomial order. (a) Odd polynomial orders. (b) Even polynomial orders.

3.3 Structural vibrations

The analysis of structural vibrations with NURBS is given a thorough treatment in [19]. Both the spectral properties of the functions as well as their application to a real world geometry are discussed in detail. Here, we examine two one-dimensional results that demonstrate the efficiency of *k*-refinement, followed by a brief discussion of the analysis of the NASA aluminum testbed cylinder.

3.3.1 Vibrations of beams and rods

We study the problem of the structural vibrations of an elastic fixed-fixed rod of unit length, whose natural frequencies and modes, assuming unit material parameters, are governed by:

$$\begin{aligned} &u_{,xx} + \omega^2 u = 0 \text{ for } x \in (0,1) \\ &u(0) = u(1) = 0, \end{aligned} \tag{3.5}$$

and for which the exact solution in terms of natural frequencies is:

$$\omega_n = n\pi, \text{ with } n = 1, 2, 3... \tag{3.6}$$

As a first numerical experiment, the eigenproblem is solved with both finite elements and isogeometric analysis using quadratic basis functions. The resulting natural frequencies, ω_n^h, are presented in Figure 3.22, normalized with respect to the exact solution (3.6), and plotted versus the

mode number, n, normalized by the total number of degrees-of-freedom, N. To produce the spectra of Figure 3.22, we used $N = 999$ but the results are in fact independent of N.

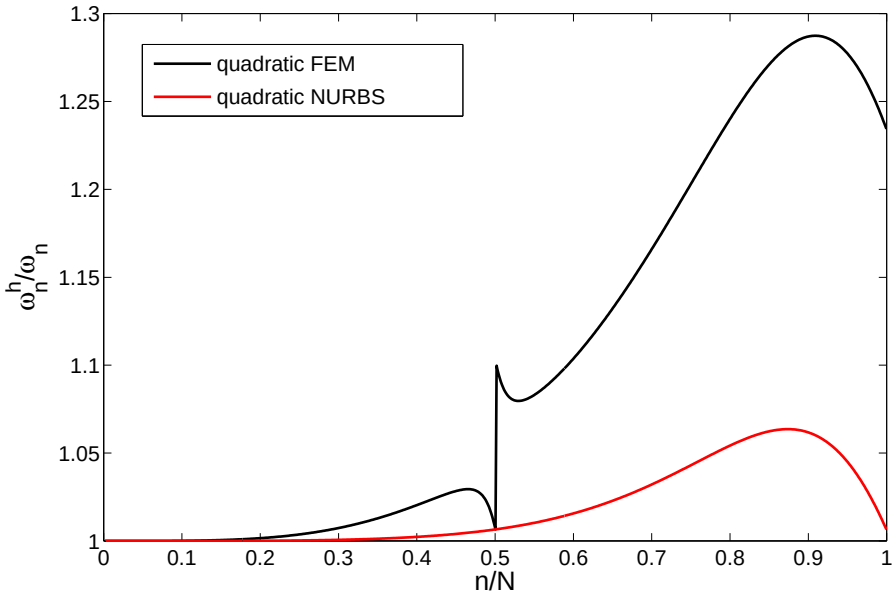


Figure 3.22: Fixed-fixed-rod. Normalized discrete spectra using quadratic finite elements and NURBS.

Figure 3.22 illustrates the superior behavior of NURBS basis functions compared with finite elements. In this case, the finite element results depict an acoustical branch for $n/N < 0.5$ and an optical branch for $n/N > 0.5$ (see Brillouin [14]). As we go to higher-order, the disparity becomes even greater. Higher-order NURBS outperform higher-order finite elements by an ever increasing margin, see Figure 3.23.

Additionally, transverse vibrations of a simply-supported, unit length Bernoulli-Euler beam are considered (see Hughes [32], Chapter 7). For this case, the natural frequencies and modes, assuming unit material and cross-sectional parameters, are governed by:

$$\begin{aligned} &u_{,xxxx} - \omega^2 u = 0 \text{ for } x \in (0,1) \\ &u(0) = u(1) = u_{,xx}(0) = u_{,xx}(1) = 0, \end{aligned} \tag{3.7}$$

where

$$\omega_n = (n\pi)^2, \text{ with } n = 1, 2, 3, ... \tag{3.8}$$

The numerical experiments and results for the Bernoulli-Euler beam problem are analogous

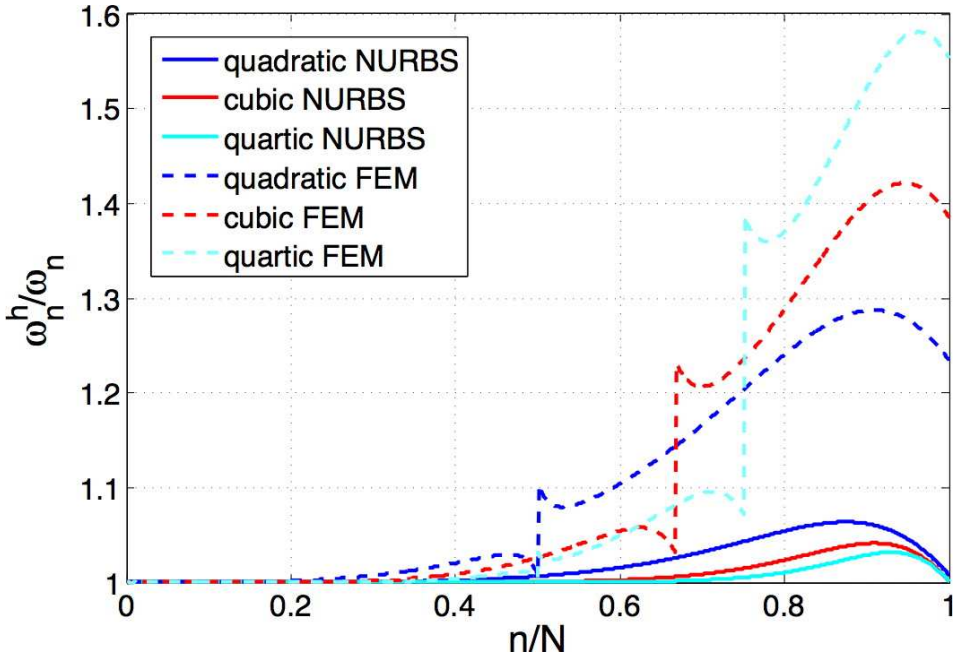


Figure 3.23: Fixed-fixed-rod. Normalized discrete spectra using higher-order finite elements and NURBS.

to the ones reported for the rod. Note that the classical beam finite element employed to solve problem (3.7) is a two-node Hermite cubic element with two degrees-of-freedom per node (transverse displacement and rotation), whereas our isogeometric analysis formulation is rotation-free (see, for example, Engel *et al.* [22]). Figure 3.24 presents the discrete spectra obtained using different order NURBS basis functions. Again, *k*-refinement results are dramatically better on a per degree-of-freedom basis.

3.3.2 NASA aluminum testbed cylinder

A less trivial example of isogeometric analysis of structural vibrations is given by the NASA aluminum testbed cylinder (ATC). The ATC is a structure inspired by the features of an airplane fuselage which is used by NASA to validate many of the modeling tools involved in the analysis and prediction of interior aircraft noise. It represents the first example of the isogeometric analysis of a "real world" geometry found in the aerospace industry, demonstrating the feasibility of constructing exact geometrical models of such complicated objects as well as the usage of NURBS on large-scale problems. More importantly, it demonstrates the profound increase in the geometrical modeling capability in simply going from linear or quadratic polynomials to *quadratic* NURBS. While higher orders may be very interesting in analysis, for the geometry this seems to be a fork in the road. It

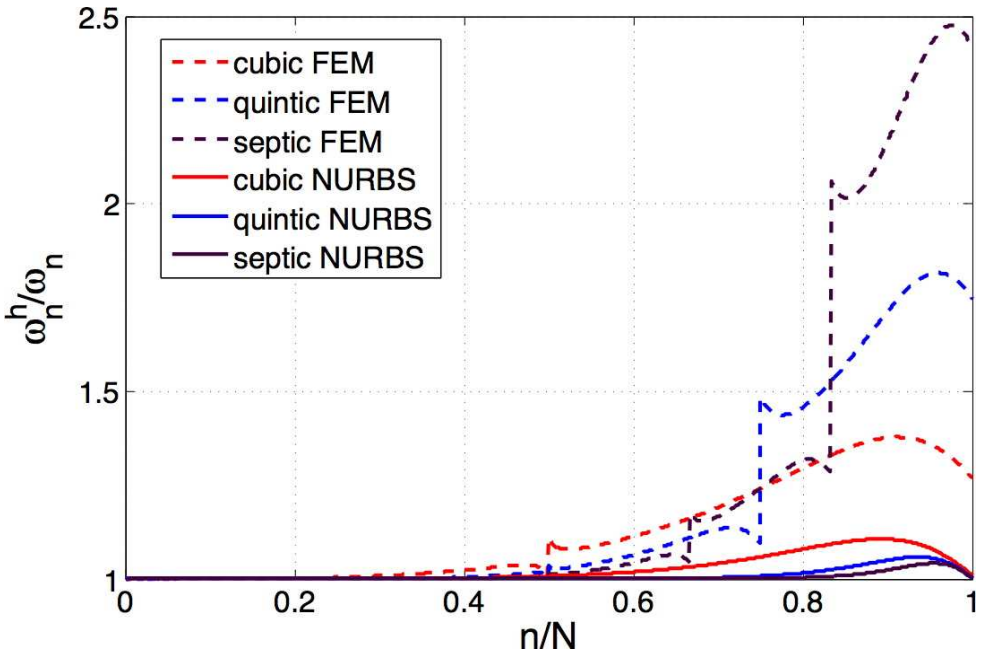


Figure 3.24: Simply-supported beam. Normalized discrete spectra using higher-order finite elements and NURBS.

is one of the major accomplishments of the research up to this point. A thorough discussion of this example is contained in [19].

The ATC is shown in Figures 3.25 and 3.26. An isogeometric model (see Figures 3.27 and 3.28) was constructed from design drawings. There are three distinct members composing the frame: nine identical main ribs (see Figures 3.29-3.33); twenty-four identical, prismatic stringers (see Figure 3.34); and two end ribs, which are mirror images of each other (see Figures 3.35-3.38). Every geometrical feature of the design drawings is *exactly* represented in the model. The stringer–main rib and stringer–end rib junctions are shown in Figures 3.39 and 3.40, respectively. Note that there are gaps between the stringer and the ribs in the notch regions.

Figure 3.25: NASA Aluminum Testbed Cylinder (ATC). Frame and skin.

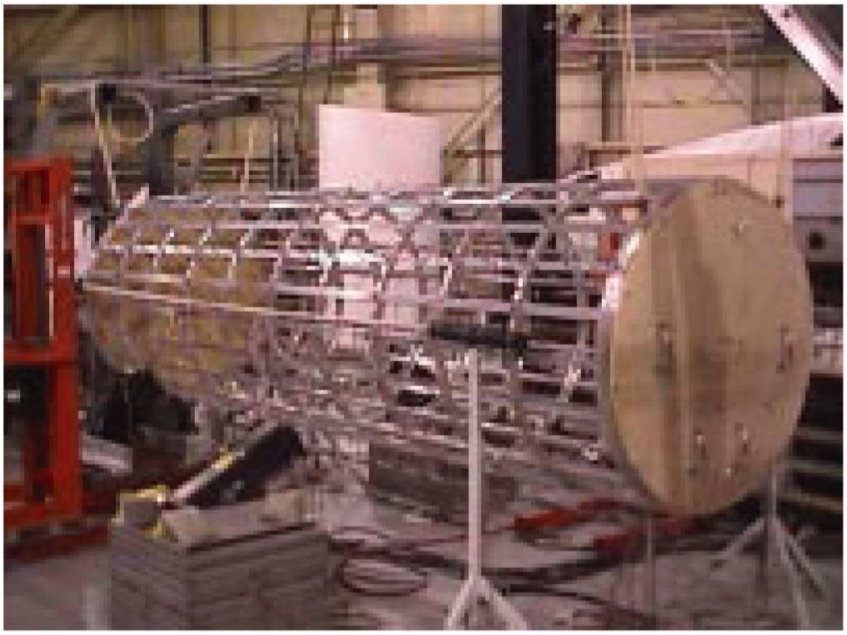


Figure 3.26: NASA ATC. Frame only.

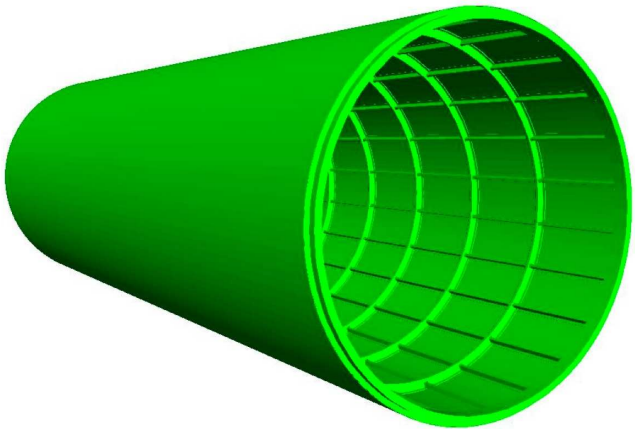


Figure 3.27: NASA ATC frame and skin: Isogeometric model.

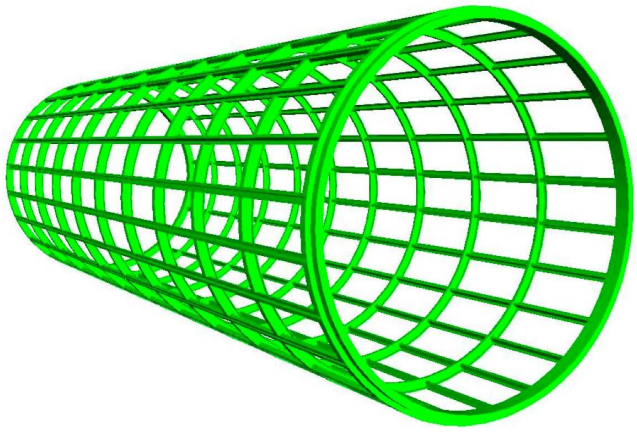


Figure 3.28: NASA ATC frame: Isogeometric model.

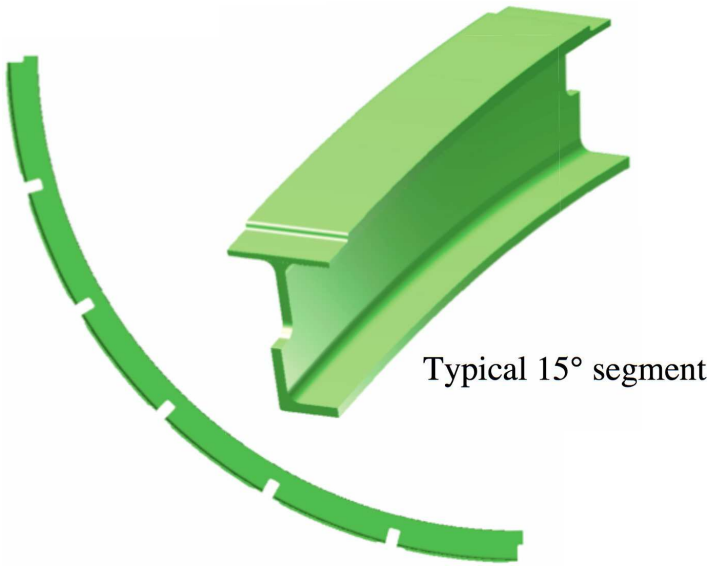


Figure 3.29: NASA ATC. Isogeometric model of the main rib.

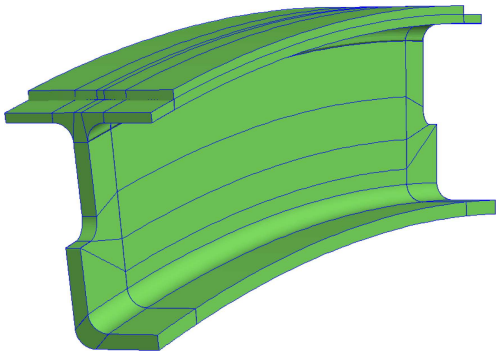


Figure 3.30: NASA ATC. Typical 15° segment of the main rib. Mesh 1, the coarsest mesh, encapsulates the exact geometry.

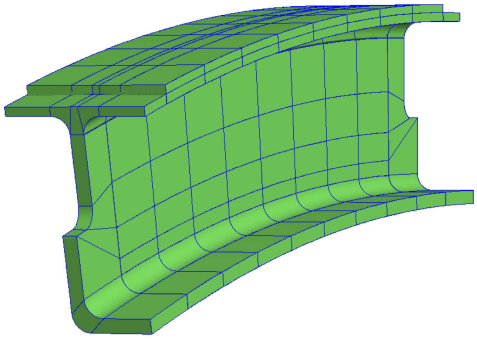


Figure 3.31: NASA ATC. Typical 15° segment of the main rib. Mesh 2. Knot insertion has been used selectively to help even out the aspect ratios of elements making Mesh 2 more uniform and suitable for analysis.

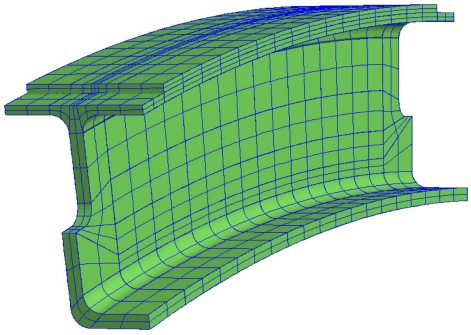


Figure 3.32: NASA ATC. Typical 15° segment of the main rib. Mesh 3. Further refinement may be necessary to resolve the solution, as is the case with standard finite element analysis, but the geometry is never altered as the mesh is refined.

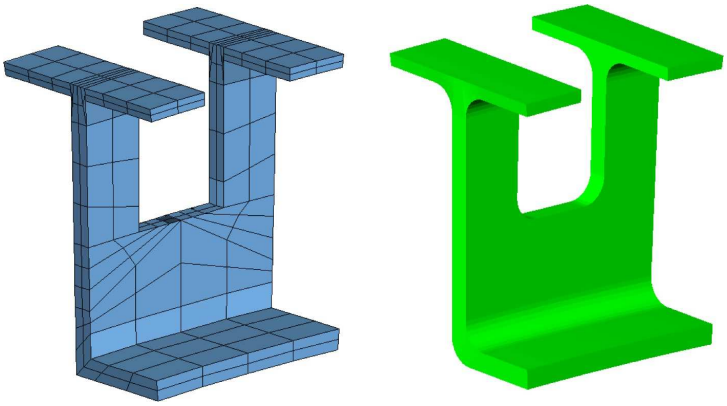


Figure 3.33: NASA ATC. Detail of the “notch” region in the main rib. The control net is on the left and the exact geometry is on the right.

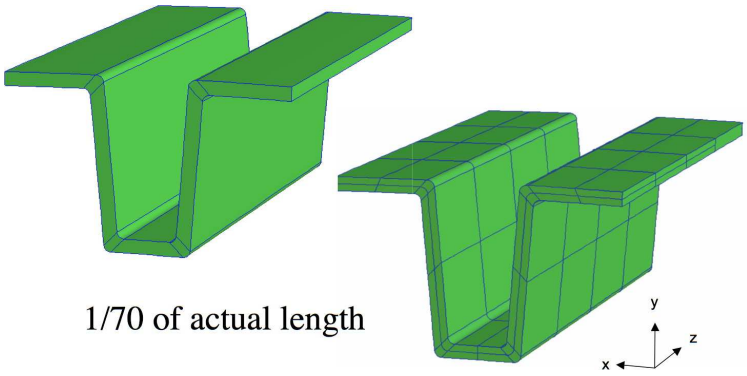


Figure 3.34: NASA ATC. Isogeometric model of the longitudinal stringer. Sample meshes.

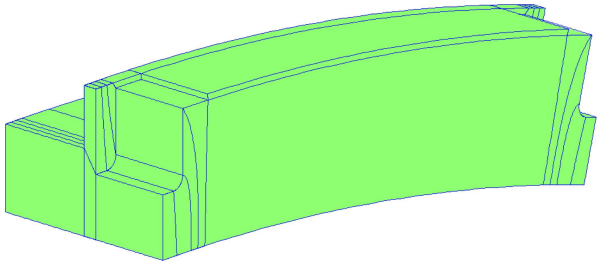


Figure 3.35: NASA ATC. Typical 15° segment of an end rib. Mesh 1 (coarsest mesh).

Experimental vibration data has been obtained for a typical isolated stringer, a typical isolated main rib, the frame assembly, and the frame and skin assembly (see [16, 20, 26] for details and reference computational results). Calculations were performed of each of the corresponding isogeometric models. The Arnoldi Package (ARPACK, see [3]) eigensolver was used in the calculations of the individual components of the ATC, while the Automated Multi-Level Substructuring (AMLS) eigensolver (see Bennighof and Lehoucq [8]) was used in the calculations of the framework and the full ATC structure.

Frequency results for the stringer are presented in Figure 3.41 and the first three bending modes are depicted in Figure 3.42. The results shown are from analysis of a single patch with one rational quadratic element in the thickness, nine rational quadratic elements through the cross-section (the smallest number capable of exactly representing the geometry), and sixteen C^5 rational sextic ($p = 6$) elements in the longitudinal direction. The resulting 144 element module has a total of 11,286 degrees of freedom. Other combinations of polynomial order, continuity and mesh size in the longitudinal direction were investigated. Higher-order, smooth meshes provided the most accuracy per degree-of-freedom but the overall run-time inevitably suffers if the polynomial order

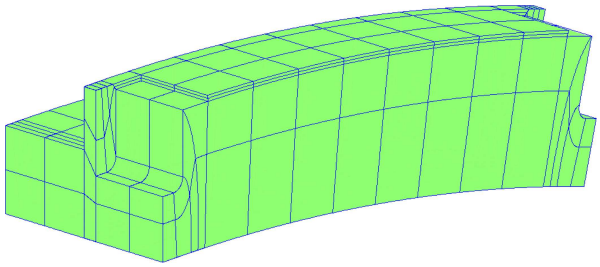


Figure 3.36: NASA ATC. Typical 15° segment of an end rib. Mesh 2.

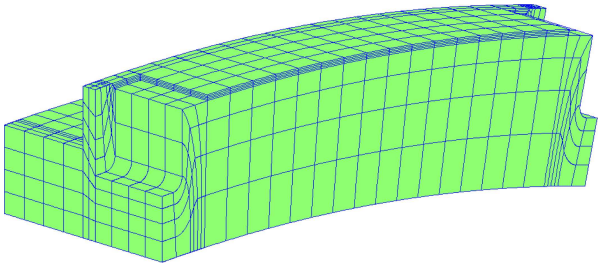


Figure 3.37: NASA ATC. Typical 15° segment of and end rib. Mesh 3.

is raised indefinitely. Further study of the myriad of refinement options offered by NURBS-based isogeometric analysis is warranted.

The main rib frequency results are presented in Figure 3.43. The "coarse mesh" (not shown) was comprised of twenty-four identical but rotated 15° sections, each comprised of six NURBS patches. Rational quadratic elements were used throughout. The full mesh for an individual rib had 34,704 degrees-of-freedom. Both *h*- and *p*-refinement were investigated for the reasons described below. In all cases, the results converged to the fine mesh results in Figure 3.43.

The isolated main rib was the only case in which some of the numerically calculated frequencies were smaller than the experimental results. The fine mesh results fall below the coarse mesh results, which for theoretical reasons must occur (see, e.g., Strang and Fix [52]). The average error for the first eight modes is larger for the fine mesh than the coarse mesh, a somewhat surprising result. Due to the upper bound property of frequencies in our formulation, and the fact that our model is geometrically exact in the sense of the design drawings, we surmise there is some discrepancy between the drawings and the as-built configuration, or some other discrepancy between the experimental configuration and our model. Nevertheless, the correlation is still reasonable. Further study is needed to determine the cause of the differences. Selected modes shapes for the fine mesh are shown in Figures 3.44 and 3.45.

Frequency results for the frame assembly are presented in Figures 3.46 and 3.47. The nu-

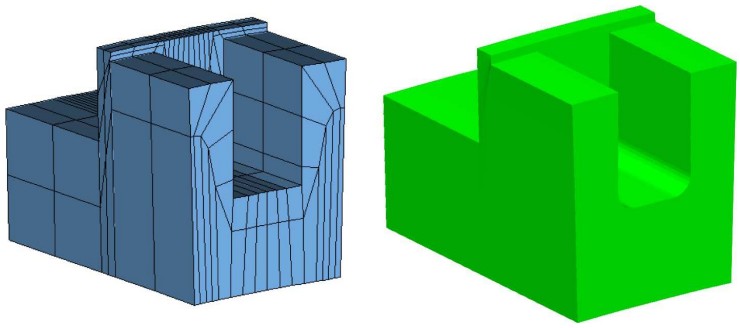


Figure 3.38: NASA ATC. Detail of the "notch" region in an end rib. The control net is on the left and the exact geometry is on the right.

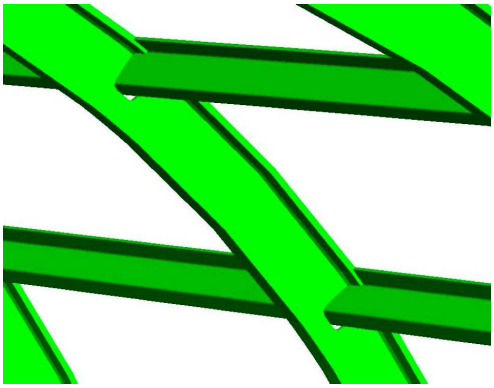


Figure 3.39: NASA ATC. Stringer–main rib junction.

merical results lie above the experimental results. The first bending and torsional modes of the frame assembly are shown in Figures 3.48-3.51. A detail of the deformation pattern in the vicinity of a main rib-stringer junction for the first torsional mode of the frame assembly is shown in Figure 3.49. A mesh of 112,200 rational quadratic elements and 1,281,528 degrees-of-freedom was used for the analysis shown. One could reduce the number of degrees-of-freedom significantly by exploiting rotational symmetry and modeling only $1/24$ of the frame assembly (as others have done, see Couchman, Dey and Barzow [20]), but part of the goal of this work was to demonstrate the feasibility of modeling an entire real structure of engineering interest using isoparametric NURBS elements, and so no such simplifications were employed.

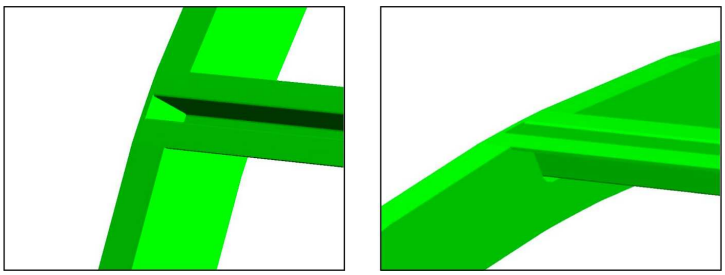


Figure 3.40: NASA ATC. Stringer–end rib junction.

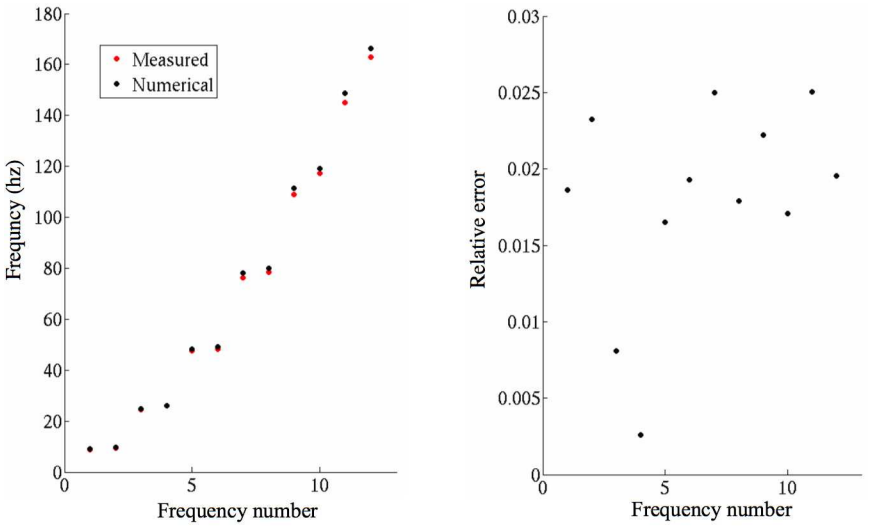


Figure 3.41: NASA ATC. Comparison of numerical and experimental frequency results for the longitudinal stringer.

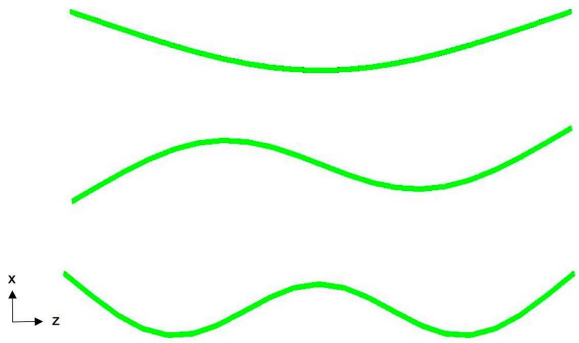


Figure 3.42: NASA ATC. Selected calculated mode shapes for the stringer. Three lowest x-z modes.

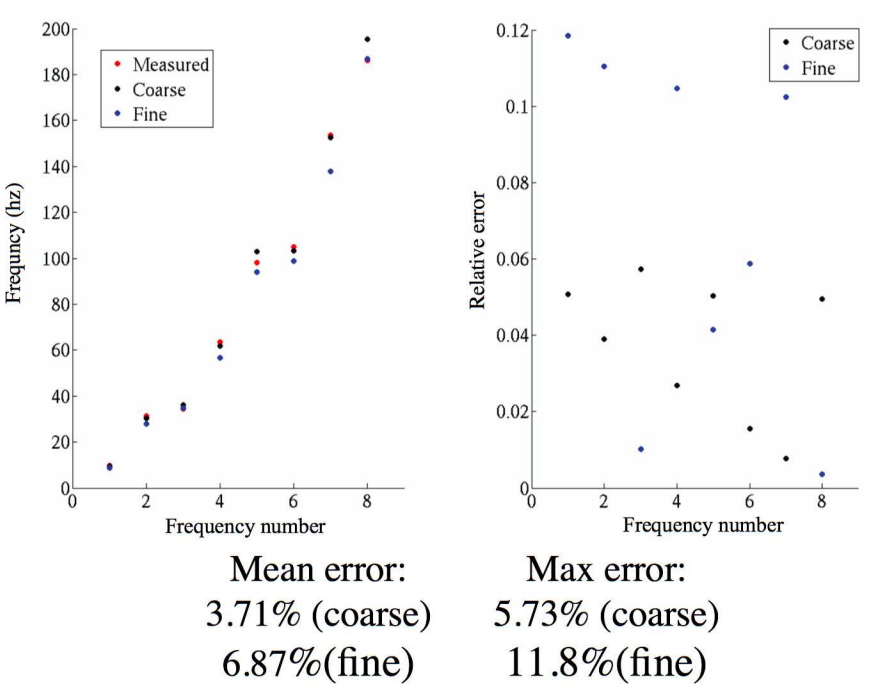


Figure 3.43: NASA ATC. Comparison of numerical and experimental frequency results for the main rib.

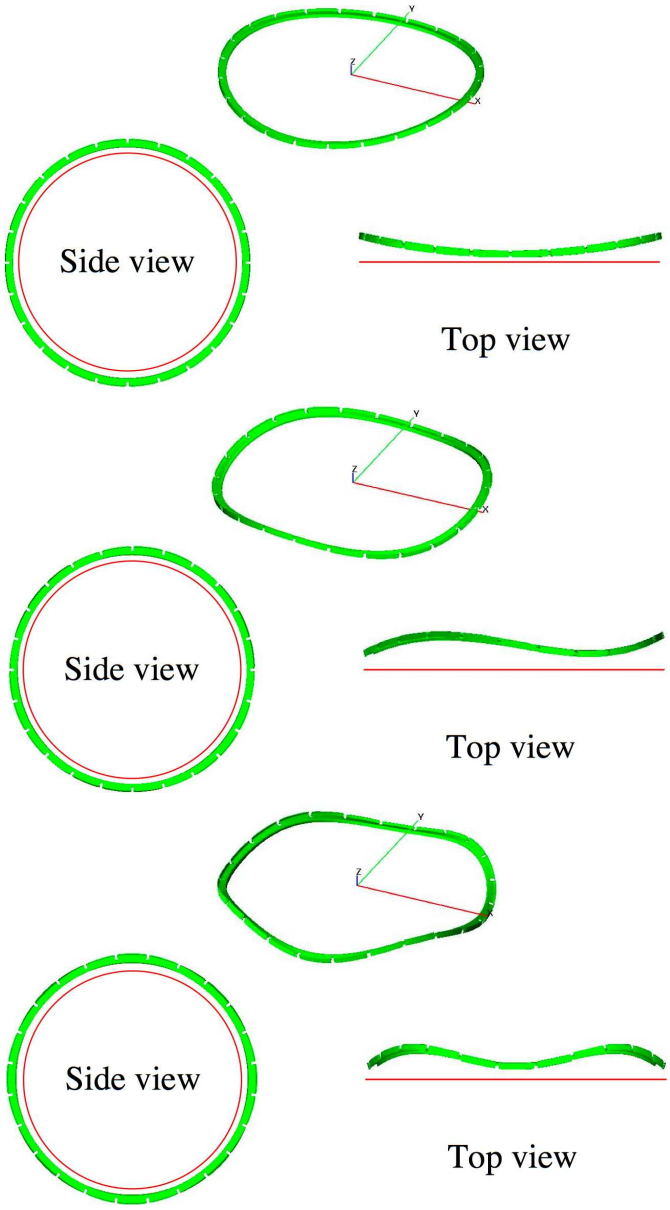


Figure 3.44: NASA ATC. Computed mode shapes for the main rib. First three out-of-plane modes.

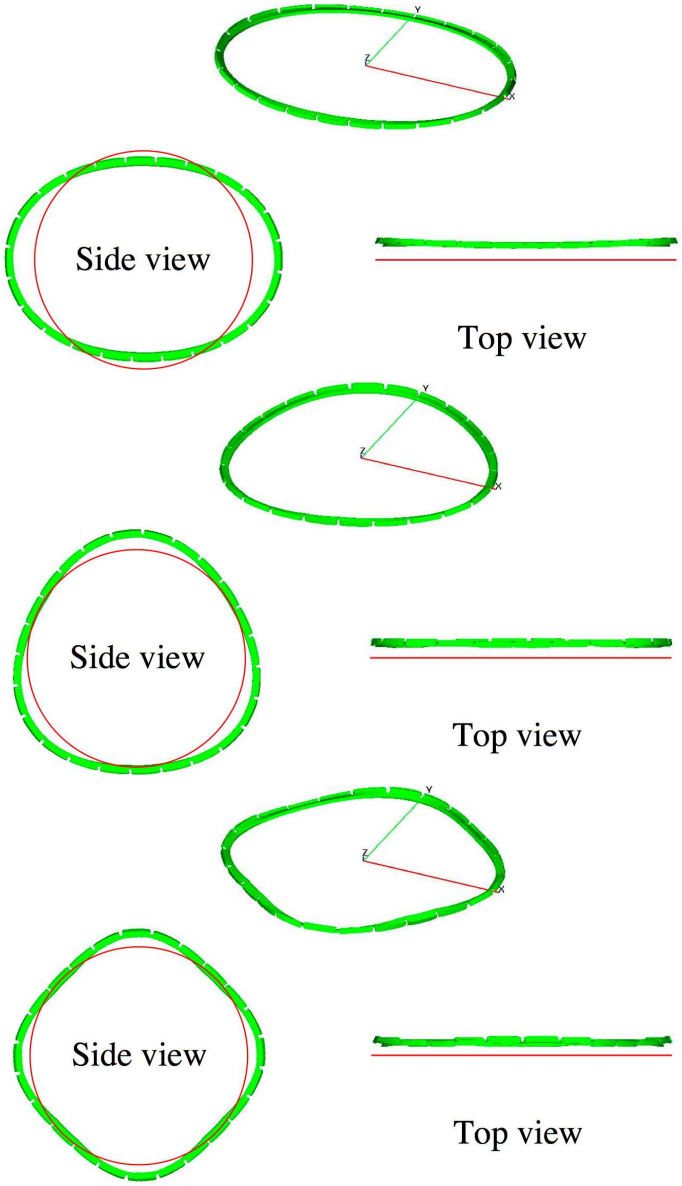


Figure 3.45: NASA ATC. Computed mode shapes for the main rib. First three in-plane modes.

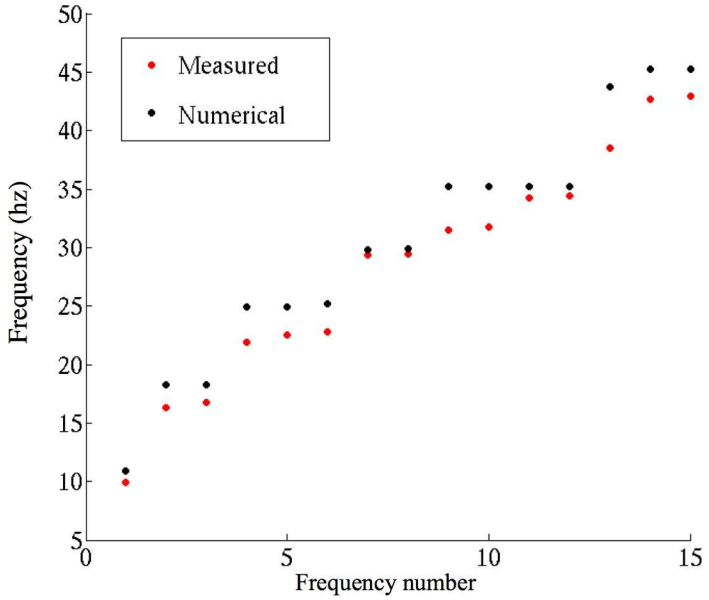


Figure 3.46: NASA ATC. Comparison of numerical and experimental frequency results for the frame assembly.

Results for the frame and skin assembly are presented in Figures 3.52 and 3.53. Once again, the numerical results lie above the experimental results. The first two modes are shown in Figures 3.54-3.56. The mesh consisted of 228,936 rational quadratic elements and 2,219,184 degrees-of-freedom. The cost of array formation and assembly was commensurate with standard quadratic finite elements.

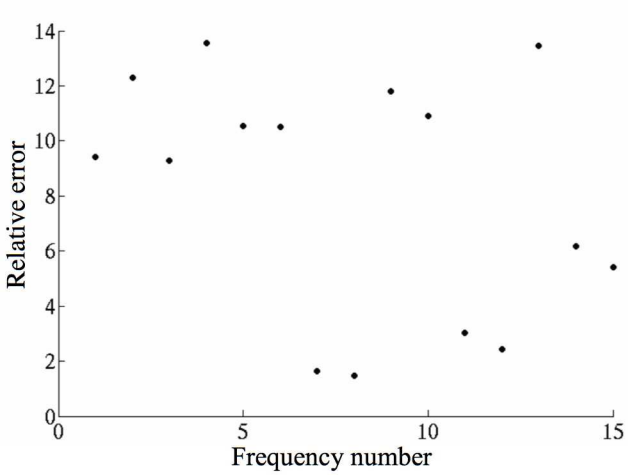


Figure 3.47: NASA ATC. Relative frequency error for the frame assembly.

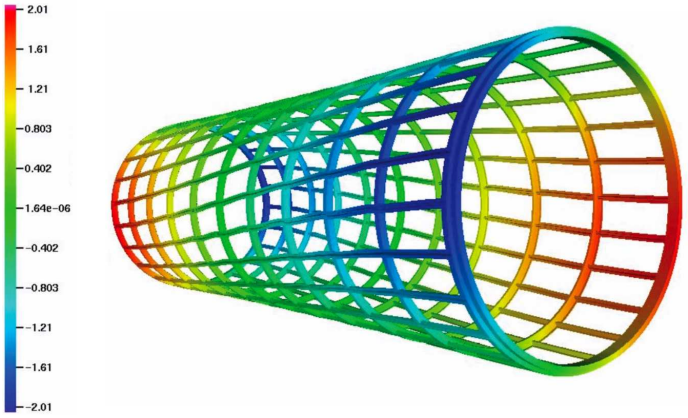


Figure 3.48: NASA ATC. Calculated first torsional mode for the frame assembly; side view. The color contours represent the vertical displacement.

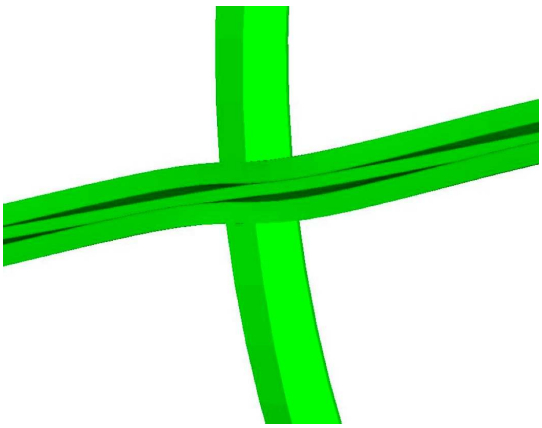


Figure 3.49: NASA ATC. Detail of first torsional mode for the frame assembly; stringer–main rib junction.

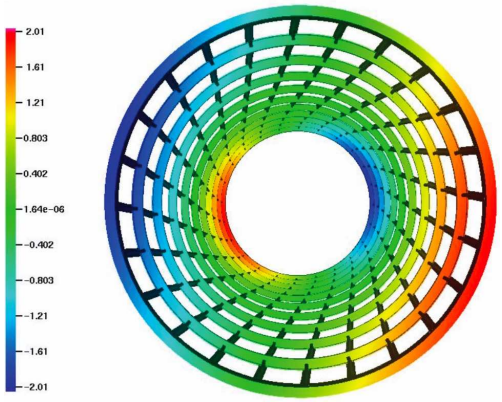


Figure 3.50: NASA ATC. Calculated first torsional mode for the frame assembly; end view. The color contours represent the vertical displacement.

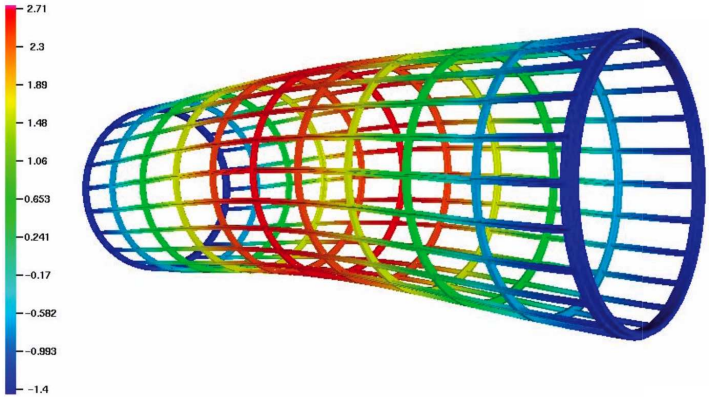


Figure 3.51: NASA ATC. Calculated first bending mode for the frame assembly. The color contours represent the vertical displacement.

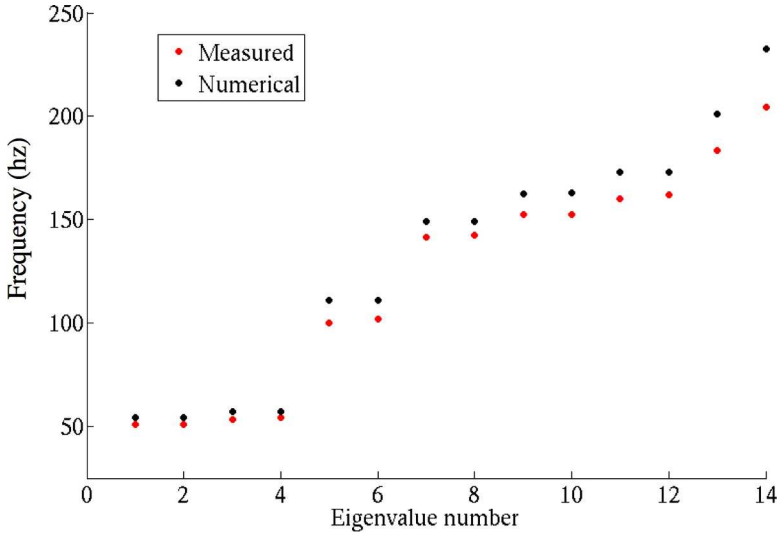


Figure 3.52: NASA ATC. Comparison of numerical and experimental frequency results for the frame and skin assembly.

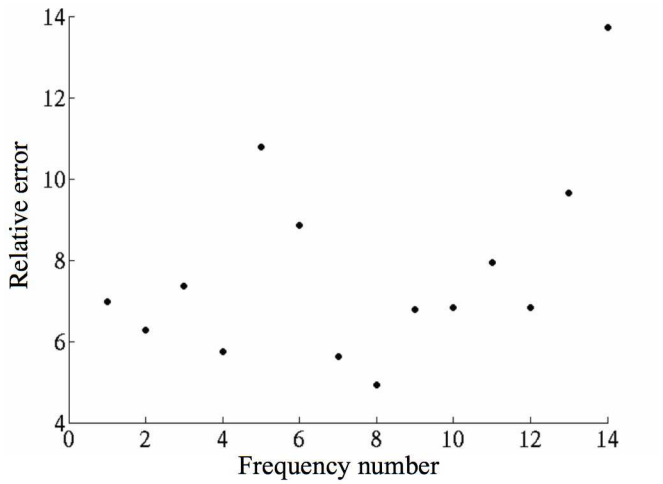


Figure 3.53: NASA ATC. Relative frequency error for the frame skin assembly.

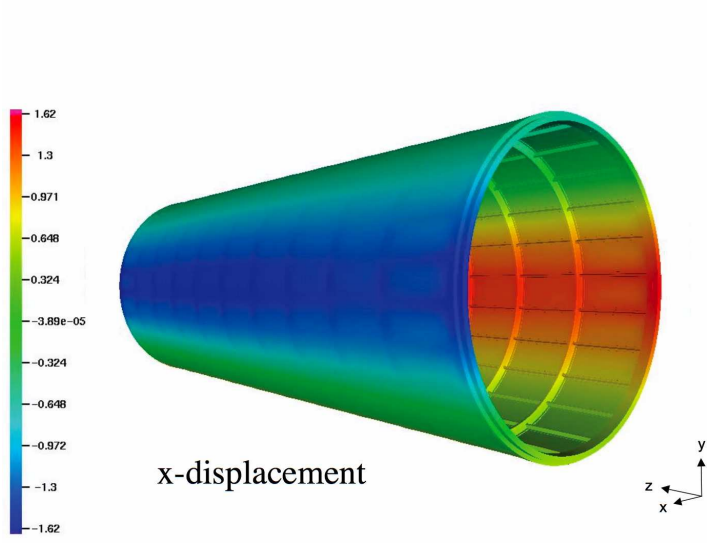


Figure 3.54: NASA ATC. Calculated first Rayleigh mode of the frame and skin assembly. The color contours represent the ovalization of the assembly.

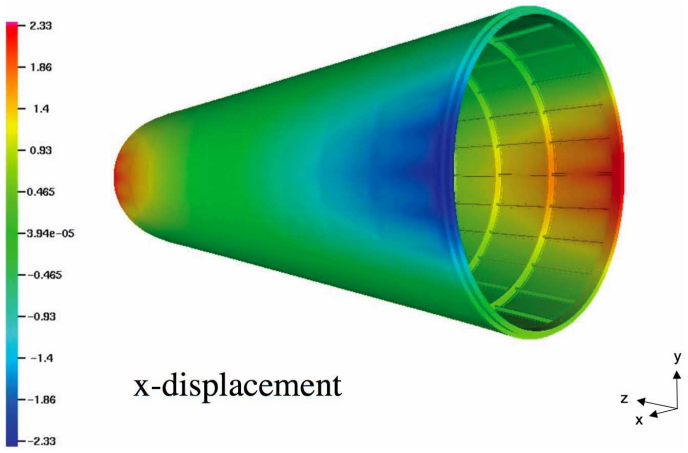


Figure 3.55: NASA ATC. Calculated first Love mode of the frame and skin assembly. The color contours represent the ovalization of the assembly.

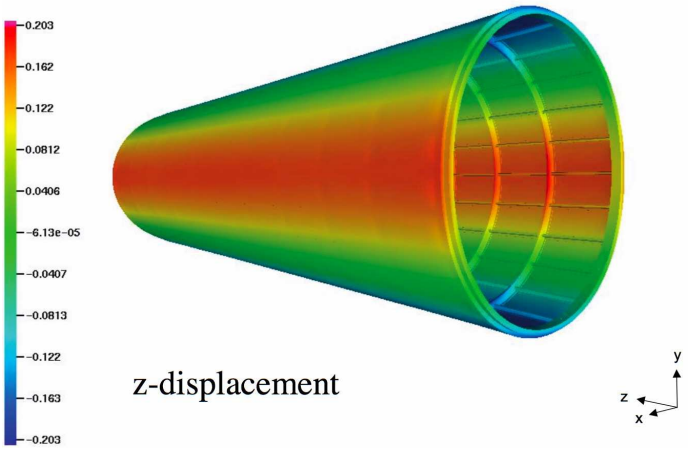


Figure 3.56: NASA ATC. Calculated first Love mode of the frame and skin assembly. The color contours represent the axial displacement of the assembly.

3.4 Current and future research in the field of isogeometric analysis

The applications currently being explored within the isogeometric analysis framework are both numerous and diverse. Problems under investigation are utilizing and expanding on all of the major features of the method. This section will present a sampling of current work in the field to give an idea of its future direction.

Several recent applications of NURBS based isogeometric analysis are benefitting from the geometrical capabilities of the method. One such example is patient specific vascular modeling for the analysis of blood flow [5, 59], see Figure 3.57. Another example is the work on structural vibrations that has been undertaken with a specific focus on naval structures as in Figures 3.58 and 3.59. Additionally, the ability of NURBS to exactly represent circles is being exploited to perform computational fluid dynamics calculations that involve rotating components such as propellers, as shown in Figures 3.60 and 3.61. The approach allows one region of the mesh to rotate within another region of the mesh, with the continuity of the solution being imposed weakly across the interface. No gaps or overlaps develop due to the exact circular geometry of the rotating region.

Additionally, work is being done both to utilize and to better understand the properties of the NURBS basis functions. The improved spectral properties of these smooth bases seem to imply the possibility for improved accuracy in applications where a wide range of modes participate in the physics. This has indeed been the case in one such application: incompressible turbulence, see [4] and [2]. Figure 3.62 shows one example of the superiority of C^1-quadratic B-spline functions over C^0-quadratic finite elements for the case of a turbulent channel flow. Analytical investigation of k-refinement is underway as well.

One of the most promising ongoing extensions of the isogeometric concept is the supplanting of the current NURBS basis with a T-spline basis. T-splines, introduced by Sederberg [49], represent a superset of NURBS. They retain the geometrical capabilities of NURBS, but allow for truly local refinement within a patch (a "T-junction" in T-splines is somewhat analogous to a "hanging node" in FEA). T-splines also offer tremendous flexibility in "sewing" together multiple patches to form large, possibly multiply-connected geometries. Bivariate and trivariate T-splines have already been implemented in an isogeometric analysis code for several linear problems, see Figure 3.63. Their extreme geometric flexibility coupled with their capacity for local refinement while maintaining high continuity appears ideally suited for use in an isogeometric analysis solver. They should offer tremendous efficiency gains over the current NURBS based codes on many types of problems.

Figure 3.57: NURBS model of patient specific abdominal aorta geometry to be used in blood flow analysis, from [5].

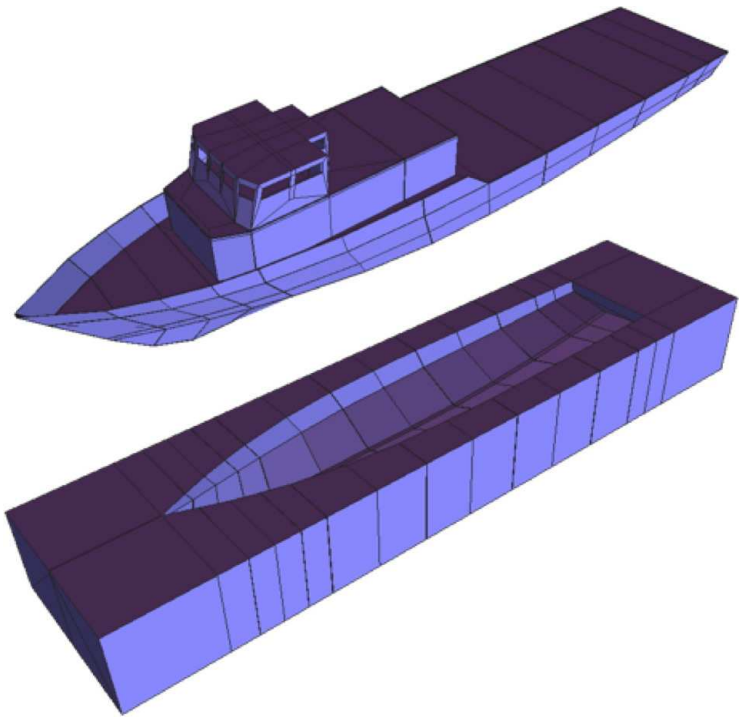


Figure 3.58: Solid NURBS mesh of a naval ship and the associated exterior mesh.

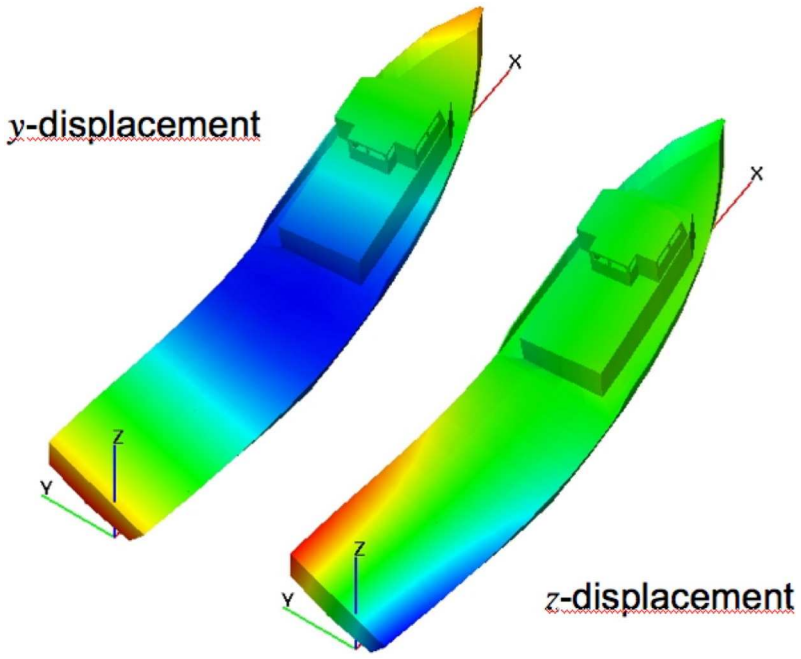


Figure 3.59: Sample vibration results of the ship from Figure 3.58.

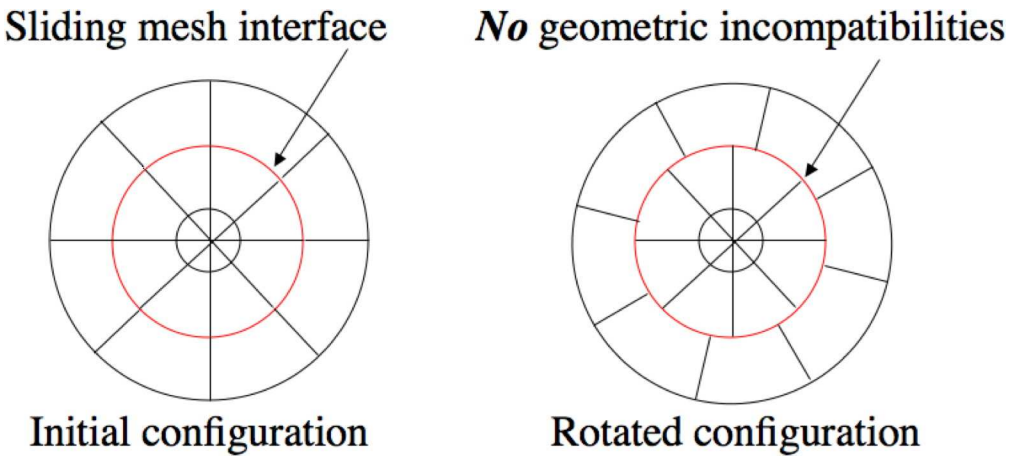


Figure 3.60: The use of NURBS allows rotating components to be accommodated by allowing one cylindrical portion of the domain to rotate within the total domain. Continuity is enforced weakly, as in a discontinuous Galerkin formulation. Finite elements cannot handle such situations without modification as rotation causes the meshes to become incompatible. Gaps and overlaps would arise due to the lack of exact circular geometries.

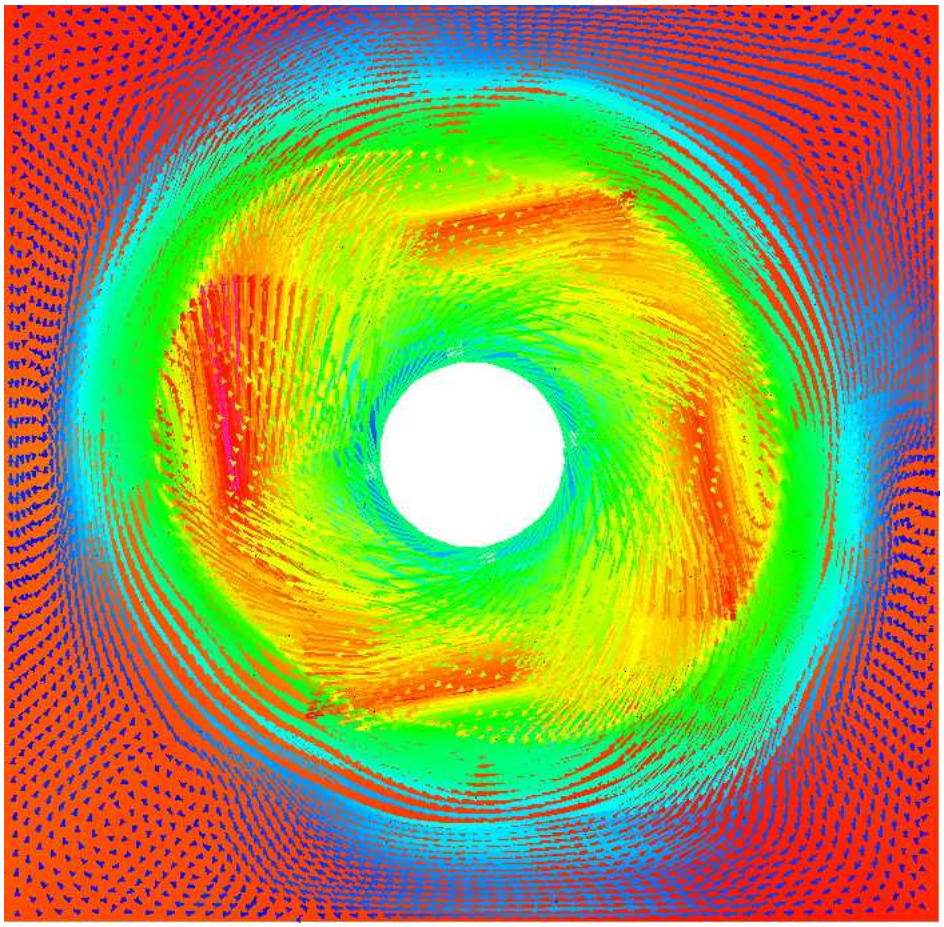


Figure 3.61: Velocity vectors and the pressure field for a rotor in a box. This 2D incompressible Navier-Stokes calculation is one of the first example calculations computed using the technique described in Figure 3.60. A circular region encompassing the rotor rotates inside the fixed mesh of the box.

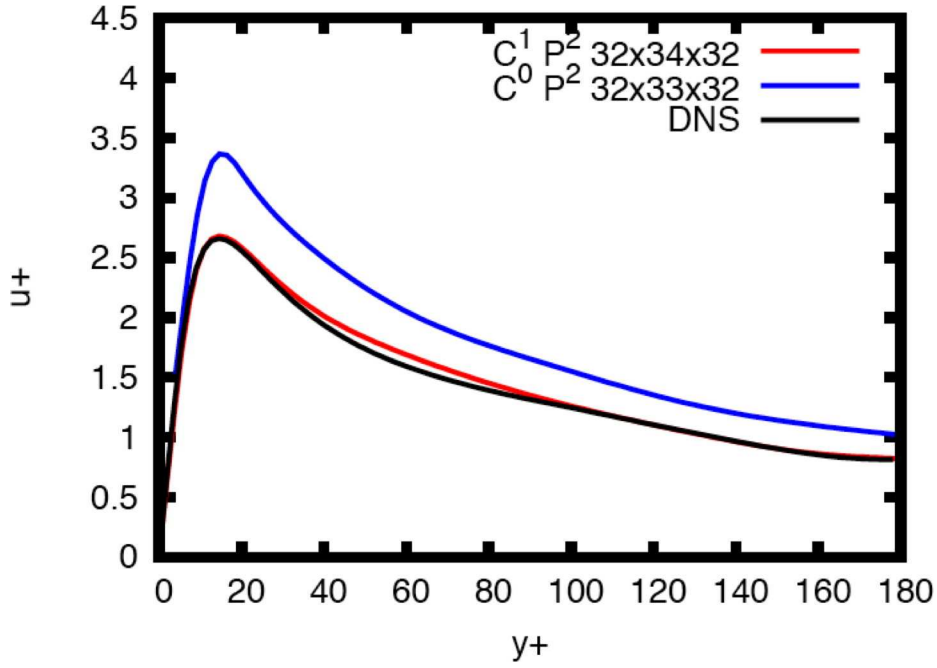
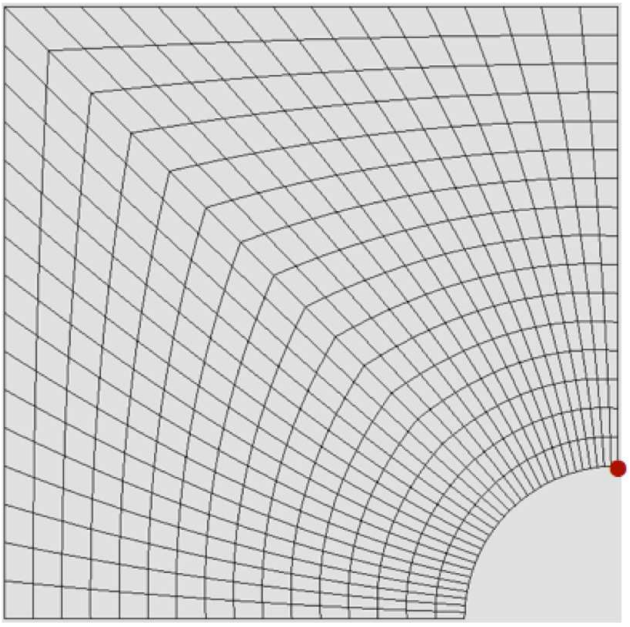
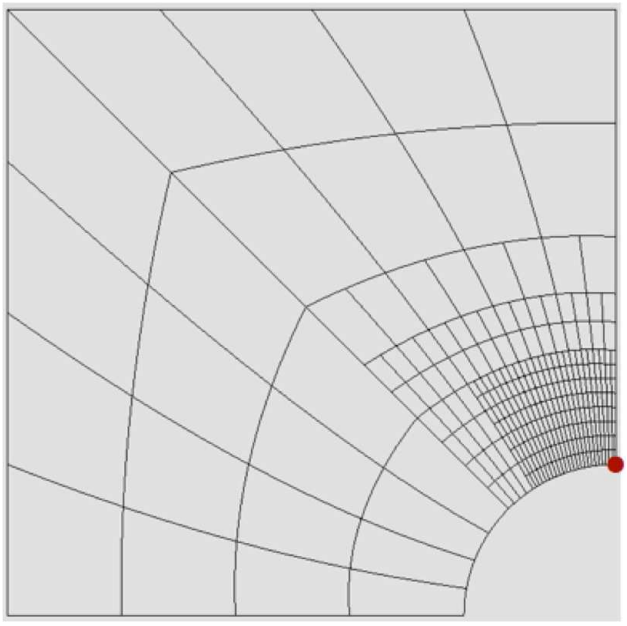


Figure 3.62: Comparison of C^0- and C^1-continuous quadratic elements with a Direct Numerical Simulation (DNS). Stream-wise velocity fluctuations shown for a turbulent channel flow at $Re_\tau = 180$ computed on a 32^3 element mesh.



(a)



(b)

Figure 3.63: Meshes for a plate with a circular hole. In both meshes, the circular hole geometry is *exactly* represented. a) A uniformly refined mesh. b) Local refinement attempts to capture a quantity of interest – in this case, the stress at the point shown in red.

Chapter 4

The Variational Multiscale Method

In this chapter we will expand on Section 1.1.2 and review the variational multiscale (VMS) method in an abstract setting. We will then examine a simple example problem to illustrate the differences between the VMS and SUPG, the first stabilized method. This will illustrate the basic components of the VMS approach as well as where difficulties arise, thus motivating the remaining chapters.

4.1 Introduction to VMS

The variational multiscale method has been an active area of research motivated by the fact that a direct application of Galerkin's method is simply not a robust approach in the presence of multiscale phenomena (see, *e.g.*, [4, 24, 31, 33, 35–37, 39, 41, 44]). As discussed in Section 1.1.2, we look at a sum decomposition of the solution $u = \bar{u} + u'$, where we identify the coarse-scale solution, $\bar{u}$, as the result of our numerical method. Traditionally, analytic expressions are sought for u'. While determining u' exactly is impossible in most situations of practical interest, we can hope to represent the *effect* of u' on $\bar{u}$ in order to produce more stable and accurate numerical solutions.

4.1.1 The abstract problem

We will introduce the variational multiscale method on an abstract linear problem, following the treatment of Hughes and Sangalli [39]. Let V be a Hilbert space with norm $\|\cdot\|_V$ generated by the inner product $(\cdot,\cdot)_V$. Additionally, we denote by V^* the dual of V and let ${}_{V^*}\langle\cdot,\cdot\rangle_V : V^* \times V \to \mathbb{R}$ be the pairing between them. Let $\mathcal{L} : V \to V^*$ represent a linear isomorphism, and let $f \in V^*$. We may consider the following problem: find $u \in V$ such that

$$\mathcal{L}u = f. \tag{4.1}$$

We may rewrite (4.1) as a variational problem: find $u \in V$ such that $\forall w \in V$

$$ {}_{V^*}\langle \mathcal{L}u, w \rangle_V = {}_{V^*}\langle f, w \rangle_V . \tag{4.2} $$

The solution to (4.2) can be expressed in terms of a Green's operator $\mathcal{G} : V^* \to V$ as $u = \mathcal{G}f$. We will refer to $\mathcal{G}$ as the ***global Green's operator*** to distinguish it from the ***fine-scale Green's operator***, $\mathcal{G}'$, introduced below.

4.1.2 The variational multiscale formulation

Let $\bar{V}$ be a closed subspace of V. We will call $\bar{V}$ the ***coarse-scale space*** and in practice we will identify it with the span of our finite element basis. Let $\mathcal{P} : V \to \bar{V}$ be a linear projector, that is, $\mathcal{P}^2 = \mathcal{P}$ and Range$(\mathcal{P}) = \bar{V}$. We define $V' = \text{Ker}(\mathcal{P})$, which is also a closed subspace of V. We will call V' the ***fine-scale space*** and identify it with the unresolved scales in our problem. With these definitions in hand, we observe that

$$ V = \bar{V} \oplus V', \tag{4.3} $$

that is, $\forall v \in V$, $\exists \bar{v}, v'$ such that $v = \bar{v} + v'$ and $\bar{v} = \mathcal{P}v \in \bar{V}$, $v' = v - \mathcal{P}v \in V'$. Specifically, we split the solution u of (4.1) as $u = \bar{u} + u'$. It is the goal of the variational multiscale method to obtain $\bar{u} = \mathcal{P}u$.

At this point we can use the linearity of the duality pairing to to split our variational problem (4.2) into

$$ {}_{V^*}\langle \mathcal{L}\bar{u}, \bar{w} \rangle_V + {}_{V^*}\langle \mathcal{L}u', \bar{w} \rangle_V = {}_{V^*}\langle f, \bar{w} \rangle_V , \quad \forall \bar{w} \in \bar{V}, \tag{4.4} $$

$$ {}_{V^*}\langle \mathcal{L}\bar{u}, w' \rangle_V + {}_{V^*}\langle \mathcal{L}u', w' \rangle_V = {}_{V^*}\langle f, w' \rangle_V , \quad \forall w' \in V'. \tag{4.5} $$

We assume that our ***coarse-scale problem*** (4.4) admits a unique solution $\bar{u} \in \bar{V}$, given u' and f. Moreover, if $\bar{V}$ is our finite element space, then it is finite-dimensional, and so if u' is given then (4.4) is tractable[1].

Analogously, we assume that the ***fine-scale problem*** (4.5) is well-posed for $u' \in V'$, given $\bar{u}$ and f. In general, V' is an infinite-dimensional space. We can express the solution to (4.5) in terms of a ***fine-scale Green's operator*** $\mathcal{G}' : V^* \to V'$, which gives an expression for u' in terms of

[1] Note that we do *not* actually need u' to be able to solve for $\bar{u}$. We only need ${}_{V^*}\langle \mathcal{L}u', \bar{w} \rangle_V$, that is, we need the *effect* of the fine scales on the coarse scales. In practice, it may be easier to model that aggregate effect than it is to find an accurate expression for u' itself.

the residual of the coarse-scale problem, that is,

$$u' = \mathcal{G}'(f - \mathcal{L}\bar{u}). \tag{4.6}$$

This allows us to completely eliminate u' from the coarse-scale problem. Inserting (4.6) into (4.4) yields the variational multiscale formulation for $\bar{u}$:

$${}_{V^*}\langle \mathcal{L}\bar{u}, \bar{w}\rangle_V - {}_{V^*}\langle \mathcal{L}\mathcal{G}'\mathcal{L}\bar{u}, \bar{w}\rangle_V = {}_{V^*}\langle f, \bar{w}\rangle_V - {}_{V^*}\langle \mathcal{L}\mathcal{G}' f, \bar{w}\rangle_V, \quad \forall \bar{w} \in \bar{V}. \tag{4.7}$$

Because of (4.3), the VMS formulation (4.7) admits a unique solution, $\bar{u} = \mathcal{P}u$.

4.1.3 The fine-scale Green's operator

The concept of the fine-scale Green's operator was introduced as an important entity in the variational multiscale method by Hughes *et al.* [33] in 1998, but it was not until the 2005 work of Hughes and Sangalli [39] that a formal expression for the operator was derived and its relationship with optimality in a given norm understood. Here we will sketch the derivation of $\mathcal{G}'$ and mention a few properties. The interested reader is referred to [39] for further details, including proofs of the well-posedness of the various steps.

Let us denote the transpose of $\mathcal{P}$ by $\mathcal{P}^T : \bar{V}^* \to V^*$, that is,

$${}_{V^*}\langle \mathcal{P}^T\bar{\mu}, w\rangle_V = {}_{\bar{V}^*}\langle \bar{\mu}, \mathcal{P}w\rangle_{\bar{V}}, \quad \forall w \in V, \bar{\mu} \in \bar{V} \tag{4.8}$$

where $\bar{V}^*$ is the dual of $\bar{V}$ and ${}_{\bar{V}^*}\langle\cdot,\cdot\rangle_{\bar{V}}$ is the pairing between them. We can rephrase the constrained fine-scale problem of (4.5) in mixed, unconstrained form using a Lagrange multiplier: find $u' \in V$, and $\bar{\lambda} \in \bar{V}^*$ such that

$$\mathcal{L}u' + \mathcal{P}^T\bar{\lambda} = r, \tag{4.9}$$

$$\mathcal{P}u' = 0, \tag{4.10}$$

where $V^* \ni r = f - \mathcal{L}\bar{u}$. As with (4.1), we can express the solution to (4.9) in terms of the global Green's operator:

$$u' = \mathcal{G}(r - \mathcal{P}^T\bar{\lambda}). \tag{4.11}$$

Applying our projector $\mathcal{P}$ to (4.11) and invoking (4.10) yields

$$\mathcal{P}\mathcal{G}r - \mathcal{P}\mathcal{G}\mathcal{P}^T\bar{\lambda} = 0, \tag{4.12}$$

and thus

$$\bar{\lambda} = (\mathcal{P}\mathcal{G}\mathcal{P}^T)^{-1}\mathcal{P}\mathcal{G}r, \tag{4.13}$$

where the invertibility of $\mathcal{P}\mathcal{G}\mathcal{P}^T$ is demonstrated in [39]. Finally, by inserting (4.13) into (4.11) we obtain

$$u' = (\mathcal{G} - \mathcal{G}\mathcal{P}^T(\mathcal{P}\mathcal{G}\mathcal{P}^T)^{-1}\mathcal{P}\mathcal{G})r. \tag{4.14}$$

Comparing (4.14) with (4.6), we arrive at an expression for the fine-scale Green's operator in terms of the global Green's operator $\mathcal{G}$ and the chosen projector $\mathcal{P}$:

$$\mathcal{G}' = \mathcal{G} - \mathcal{G}\mathcal{P}^T(\mathcal{P}\mathcal{G}\mathcal{P}^T)^{-1}\mathcal{P}\mathcal{G}. \tag{4.15}$$

Two properties of the fine-scale Green's operator immediately follow from (4.15):

$$\mathcal{G}'\mathcal{P}^T = 0, \tag{4.16}$$

$$\mathcal{P}\mathcal{G}' = 0. \tag{4.17}$$

Remark 4.1.1. *In what follows, we will always consider $\mathcal{P}$ to be an orthogonal projector. Given an inner product $(\cdot,\cdot)$ defined on $V \times V$, possibly different than $(\cdot,\cdot)_V$, the associated orthogonal projector $\mathcal{P}$ is defined by*

$$(\mathcal{P}w, \bar{v}) = (w, \bar{v}), \quad \forall w \in V, \forall \bar{v} \in \bar{V}. \tag{4.18}$$

In this setting, the result $\bar{u} \in \bar{V}$ of our VMS formulation will be optimal with respect to the norm $\|\cdot\|$ induced by the inner product $(\cdot,\cdot)$. That is,

$$\|u - \bar{u}\| \leq \|u - \bar{v}\|, \quad \forall \bar{v} \in \bar{V}. \tag{4.19}$$

4.2 VMS and SUPG: An example in one dimension

Though a thorough review of stabilized methods is beyond scope of this work (see, *e.g.,* [36]), a simple example problem will demonstrate the superiority of the variational multiscale method over classical stabilized methods in an idealized setting. Simultaneously, it will expose some the the barriers to the practical use of pure VMS in many situations. This will serve to motivate the research of the next section.

The Streamline Upwind Petrov-Galerkin (SUPG) method was introduced by Brooks and Hughes [15]. The goal of SUPG was to suppress the spurious oscillations associated with a straightforward application of Galerkin's method to many classes of advection-dominated phenomena with-

out upsetting the consistency of the method and with a minimal adverse impact on accuracy. It was the first consistent stabilized method, and has in many ways been the most successful, finding its way into many commercially available and industrial codes. Though other stabilized methods exist, we will use SUPG as the prototypical example of a stabilized method prior to the advent of VMS.

The creation of SUPG followed an "engineering approach," that is, the method was not *derived* but, rather, *designed* to have certain desirable properties. It results in a pointwise exact solution for advection-diffusion in 1D with a constant advective velocity and a simple forcing function when linear elements are used [15]. Interestingly, it has been show in [31] that SUPG is equivalent to VMS in this case. We need only consider a slightly more complicated example to observe the differences between the methods.

4.2.1 1D example with non-constant velocity

Let us consider the following advection-diffusion[2] example on a domain Ω in one dimension:

$$a(x)u_{,x} - \kappa u_{,xx} = f \quad \text{in } \Omega, \tag{4.20}$$

$$u = g_b \quad \text{on } \partial\Omega, \tag{4.21}$$

where we have explicitly written the advective velocity $a = a(x)$ in (4.20) to show its spatial dependence. Henceforth, we will not explicitly write the dependence on x in our notation for the sake of brevity. The diffusivity, κ, is assumed constant. Concisely, (4.20) is simply $\mathcal{L}u = f$, where $f \in L^2(\Omega)$ and

$$\mathcal{L} = a\frac{d}{dx} - \kappa\frac{d^2}{dx^2}. \tag{4.22}$$

We may rewrite (4.20) and (4.21) as a variational problem: find $u \in V = H_0^1(\Omega) + g_b$ such that

$$(w, au_{,x})_\Omega + (w_{,x}, \kappa u_{,x})_\Omega = (w, f)_\Omega \quad \forall w \in V_0 = H_0^1(\Omega), \tag{4.23}$$

where $(\cdot,\cdot)_\Omega$ is the L^2 inner product on Ω. We will consider four approaches to solving this problem. The first two approaches are variants of VMS, the third is Galerkin's method, and the fourth is SUPG.

[2] This is a variation on the classical advection-diffusion equation in that we do not assume a divergence free velocity field.

VMS

The VMS approach begins with the exact variational statement of the problem given in (4.23). As in Section 4.1.2, we consider a splitting of our solution space[3] $V = \bar{V} \oplus V'$ and likewise with our weighting space $V_0 = \bar{V}_0 \oplus V'_0$. By linearity with respect to the weighting function, we have the following two subproblems:

$$(\bar{w}, a(\bar{u}_{,x} + u'_{,x}))_\Omega + (\bar{w}_{,x}, \kappa(\bar{u}_{,x} + u'_{,x}))_\Omega = (\bar{w}, f)_\Omega \quad \forall \bar{w} \in \bar{V}_0, \tag{4.24}$$

$$(w', a(\bar{u}_{,x} + u'_{,x}))_\Omega + (w'_{,x}, \kappa(\bar{u}_{,x} + u'_{,x}))_\Omega = (w', f)_\Omega \quad \forall w' \in V'_0. \tag{4.25}$$

After integrating by parts (recalling that a is dependent on x) we can rewrite our coarse-scale equation (4.24) as

$$(\bar{w}, a\bar{u}_{,x})_\Omega + (\bar{w}_{,x}, \kappa\bar{u}_{,x})_\Omega + (\mathcal{L}^*\bar{w}, u')_\Omega = (\bar{w}, f)_\Omega \quad \forall \bar{w} \in \bar{V}_0, \tag{4.26}$$

where, $\mathcal{L}^*\bar{w} = -a\bar{w}_{,x} - a_{,x}\bar{w} - \kappa\bar{w}_{,xx}$ is the adjoint of $\mathcal{L}$. Given u', (4.26) is an exact equation for $\bar{u}$.

If we restrict ourselves to the special case of linear elements[4], it is well known (see [12, 13, 31, 33]) that H^1-optimality results in a nodally exact solution in this one-dimensional setting. This means that not only is $u'|_{\partial\Omega} = 0$, but the fine-scale solution vanishes on the element boundaries as well. Thus, in seeking u', we may restrict our fine-scale problem (4.25) to an individual element Ω_e. Doing so, and simplifying notation, (4.25) becomes: find $u' \in H^1_0(\Omega_e)$ such that

$$(w', \mathcal{L}u')_{\Omega_e} = -(w', \mathcal{L}\bar{u} - f)_{\Omega_e} \quad \forall w' \in H^1_0(\Omega_e), \tag{4.27}$$

for $e = 1, \ldots, n_{el}$, where n_{el} is the number of elements in the mesh. By restricting our fine-scale problem to the element in this way, we have in fact chosen our projector $\mathcal{P}$ from Section 4.1.3 to be the H^1-projector. We may now obtain the fine-scale Green's operator associated with (4.25) directly, without the construction of (4.15), by finding the ***element Green's operator*** associated with (4.27). In this context, it is convenient to represent the Green's operator $\mathcal{G}$ through the Green's function $g : \Omega \times \Omega \to \mathbb{R}$ such that

$$u(y) = \int_\Omega g(x, y) f(x) dx \quad \forall y \in \Omega. \tag{4.28}$$

[3] We will assume the boundary conditions are accounted for in the coarse-scale space. That is, $\bar{V} \subset H^1_0(\Omega) + g_b$ and $V' \subset H^1_0(\Omega)$.

[4] Linear elements in one dimension are a *very* special case. The approach taken in this section to obtain $\mathcal{G}'$ does not generalize. Of course, the standard approach of Section 4.1.3 could be used here instead.

Taking into account the localizing effect of H^1-projection in this one-dimensional context, we note that the element Green's function and the fine-scale Green's function, restricted to the element Ω_e, are identical. We have the following problem for the element Green's function g^e:

$$\mathcal{L}^* g^e(x,y) = \delta(x-y) \quad \text{in } \Omega_e, \tag{4.29}$$

$$g^e(x,y) = 0 \quad \text{on } \partial\Omega_e, \tag{4.30}$$

for $e = 1, \ldots, n_{el}$. Once this is solved analytically, the fine-scale solution is given by

$$u'(y) = \int_{\Omega_e} g^e(x,y)\,(f(x) - \mathcal{L}\bar{u}(x))\,dx \quad \forall y \in \Omega_e. \tag{4.31}$$

Finally, (4.31) can be inserted back into (4.26) to solve for $\bar{u}$ exactly. We will denote this approach by "VMS-g'" as it uses the fine-scale Green's function $g' = g^e$ to solve for u'.

In cases where it is impossible or impractical to obtain g', it is common practice to follow the example of stabilized methods and to make the approximation

$$u' \approx -\tau \cdot (\mathcal{L}\bar{u} - f), \tag{4.32}$$

(see [31]). We will call this approach "VMS-τ." Letting $\bar{V} = V^h$, $\bar{V}_0 = V_0^h$, and substituting (4.32) into (4.26) yields

$$\begin{aligned}(w^h, a{u^h}_{,x})_\Omega + ({w^h}_{,x}, \kappa u^h_{,x})_\Omega + \\ (a{w^h}_{,x}, \tau(\mathcal{L}u^h - f))_{\Omega^{int}} + (a_{,x}w^h, \tau(\mathcal{L}u^h - f))_{\Omega^{int}} + (\kappa {w^h}_{,xx}, \tau(\mathcal{L}u^h - f))_{\Omega^{int}} \\ = (w^h, f)_\Omega \quad \forall w \in V_0^h,\end{aligned} \tag{4.33}$$

Remark 4.2.1. *It is important to note the fact that the coarse-scale equations for VMS-τ and VMS-g' are exactly the same, the only difference is that in VMS-τ we are approximating u' using (4.32), while in VMS-g' we are solving for u' analytically using (4.31).*

Many papers have been written on the selection of the stabilization parameter τ (see, *e.g.*, [1, 28]), but the majority of the approaches are based on scaling arguments of some type – τ is selected, not derived. The basic argument for such an approach goes as follows (see [33]): Consider rewriting (4.31) in terms of an integral operator $\mathcal{G}'$ (see (4.6)) as

$$u' = -\mathcal{G}'(\mathcal{L}\bar{u} - f). \tag{4.34}$$

If g' or analogously $\mathcal{G}'$ are prohibitively difficult to obtain or to use, we could approximate the

integral operator $\mathcal{G}'$ by an algebraic operator τ that scales with the flow parameters in the same way as $\mathcal{G}'$. The approximation $\tau \approx \mathcal{G}'$ leads directly to (4.32).

Galerkin's method

Galerkin's method seeks a solution to (4.23) in a finite-dimensional subspace V^h that is spanned by the finite element basis. The Galerkin problem is simply: find $u^h \in V^h \subset H_0^1(\Omega) + g_b$ such that

$$(w^h, au^h_{,x})_\Omega + (w^h_{,x}, \kappa u^h_{,x})_\Omega = (w^h, f)_\Omega \quad \forall w \in V_0^h \subset H_0^1(\Omega). \tag{4.35}$$

Though Galerkin's method substantially predates VMS, it may be interpreted in the multiscale context by considering the splitting into coarse and fine scales. Letting $\bar{u} = u^h$, (4.35) is simply the coarse-scale equation (4.26) under the assumption that $u' \equiv 0$.

SUPG

The SUPG formulation begins with the Galerkin formulation (4.35), and adds an additional term to attempt to improve stability. It reads, find $u^h \in V^h \subset H_0^1(\Omega) + g_b$ such that

$$\begin{aligned}(w^h, au^h_{,x})_\Omega + (w^h_{,x}, \kappa u^h_{,x})_\Omega + & \\ & (aw^h_{,x}, \tau(\mathcal{L}u^h - f))_{\Omega^{int}} \\ & = (w^h, f)_\Omega \quad \forall w \in V_0^h \subset H_0^1(\Omega).\end{aligned} \tag{4.36}$$

The additional term on the middle line of (4.36) consists of *only the advective part* of the linear operator, $\mathcal{L}_{adv} = a\frac{d}{dx}$, acting on the weighting function, times a scaling parameter τ multiplying the residual $\mathcal{L}u^h - f$, all integrated over the element interiors (denoted Ω^{int}). This weighted residual form of the stabilization term guarantees consistency, that is, the exact solution u satisfies (4.36).

Compare (4.36) to the VMS-τ formulation of the problem (4.33), concentrating on the middle line of each. They are very similar, but with two additional terms in (4.33). With linear elements, the term involving $u_{,xx}$ is identically zero, but the expression involving the spatial derivative of the velocity $a_{,x}$ is a nontrivial difference between VMS-τ and SUPG. Despite the consistency of the method, SUPG has not been derived and does not admit any specific interpretation in the sense of VMS, as did Galerkin's method. This has been one of its biggest criticisms, as well as one of the biggest barriers to its improvement.

4.2.2 Numerical results

For concreteness, let us choose $a(x) = -x$ and consider domain $\Omega = (-1, 1)$. Equations (4.20) and (4.21) become

$$\begin{aligned} -xu_{,x} - \kappa u_{,xx} &= 0 \quad \text{on } \Omega, && (4.37)\\ u(-1) &= -1, && (4.38)\\ u(1) &= 1. && (4.39) \end{aligned}$$

The exact solution to this problem may be found analytically,

$$u(x) = \text{erf}\left(\frac{\sqrt{2}}{2\sqrt{\kappa}}x\right) / \text{erf}\left(\frac{\sqrt{2}}{2\sqrt{\kappa}}\right), \tag{4.40}$$

where

$$\text{erf}(z) = \frac{2}{\sqrt{\pi}} \int_0^z e^{-t^2} dt. \tag{4.41}$$

Figure 4.1 shows the exact solution plotted for several different values of κ.

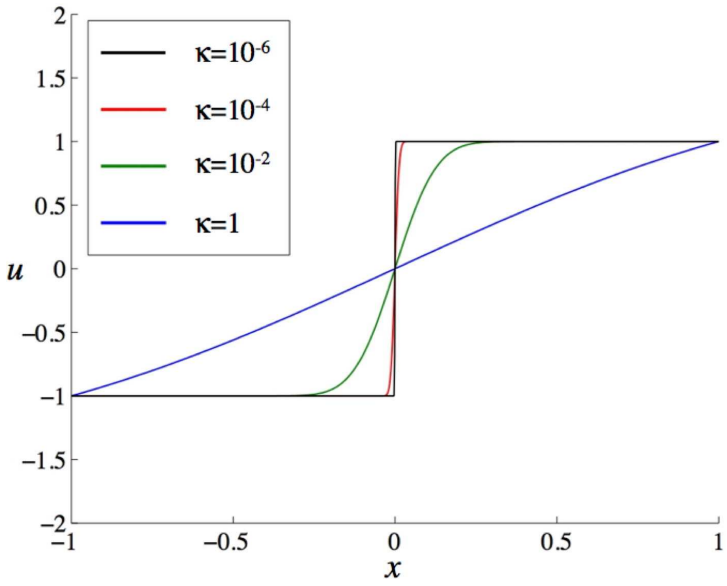


Figure 4.1: As κ decreases, the problem becomes more advection-dominated and the layer becomes sharper. Very sharp layers engender instabilities on meshes too coarse to resolve them.

In such a simple setting, we can find the element Green's function, g^e, analytically. Specifi-

cally,

$$g^e = \begin{cases} -\beta \frac{\sqrt{\pi} e^{\beta^2 x^2} \left(\text{erf}(\beta y) - \text{erf}(\beta x_{i+1})\right)\left(\text{erf}(\beta x_i) - \text{erf}(\beta x)\right)}{\text{erf}(\beta x_{i+1}) - \text{erf}(\beta x_i)} & x \leq y, \\ -\beta \frac{\sqrt{\pi} e^{\beta^2 x^2} \left(\text{erf}(\beta y) - \text{erf}(\beta x_i)\right)\left(\text{erf}(\beta x_{i+1}) - \text{erf}(\beta x)\right)}{\text{erf}(\beta x_{i+1}) - \text{erf}(\beta x_i)} & y < x, \end{cases} \tag{4.42}$$

where $\beta = 1/\sqrt{2\kappa}$ and x_i is the i^{th} node, $i = 1, \ldots, N$.

We will take $\kappa = 10^{-3}$, which will clearly make the problem advection dominated in all but a tiny neighborhood around $x = 0$. The problem will be solved using the various formulations discussed thus far on a uniform mesh comprised of 20 linear elements. Figure 4.2 shows individual results for each of the methods discussed, while Figure 4.3 plots them on the same axis, together with the exact solution for comparison. The Galerkin's method solution shows large oscillations. The SUPG solution shows much better stability but it is still not monotone. The VMS-τ solution has roughly the same pointwise accuracy as the SUPG solution but it is monotone. Heuristically speaking, its solution would be preferable to the non-monotone solution of SUPG. This is in part because many situations exist in which a non-monotone solution can lack any physical significance (*e.g.* negative temperatures or densities, concentrations greater that one, etc.), while the monotone solution may be overly diffusive, but not meaningless. Another reason to prefer monotone solutions is because in nonlinear applications, over-shoots and under-shoots can lead to a lack of convergence of the iterative solver. A more stable solution with approximately the same accuracy is more "fail-safe." Lastly, the VMS-g' solution interpolates the solution at the nodes, as expected.

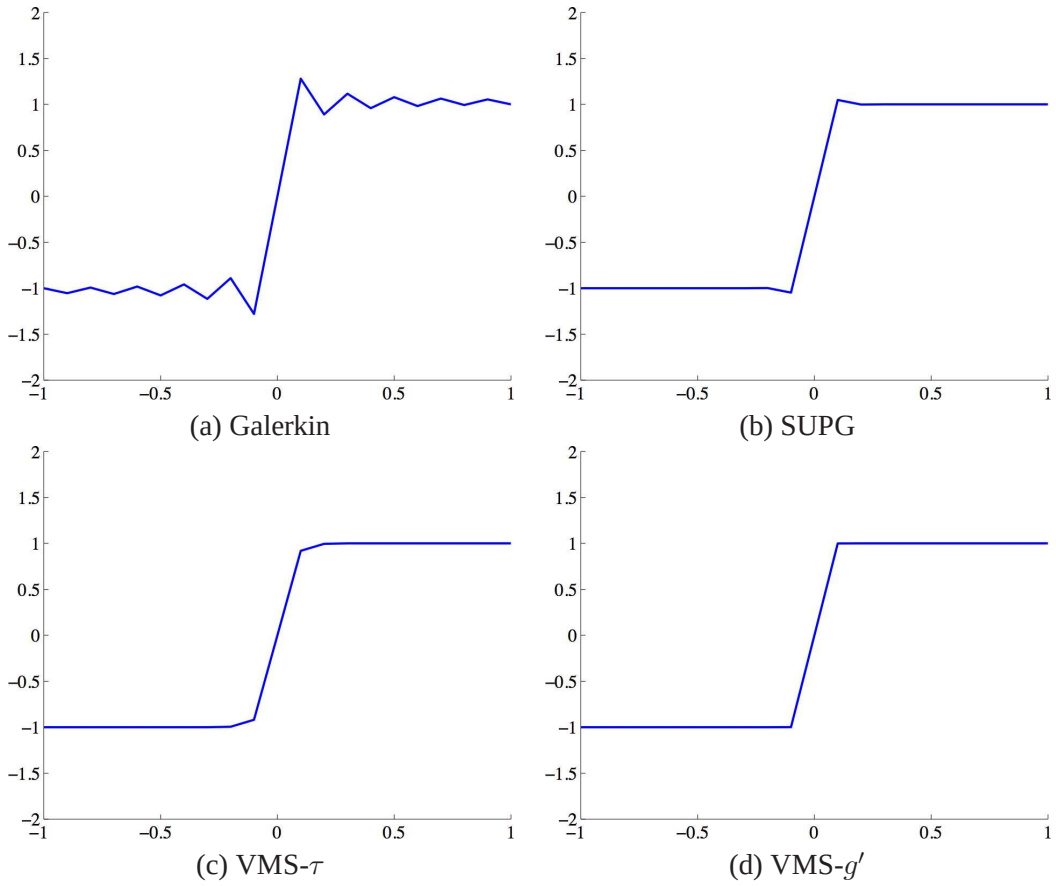


Figure 4.2: Results for advection-diffusion with non-constant velocity. a) The Galerkin solution has large spurious oscillations. b) The SUPG solution is much more stable but non-monotone. c) The VMS-τ solution is stable and monotone. d) The VMS-g' solution is nodally exact.

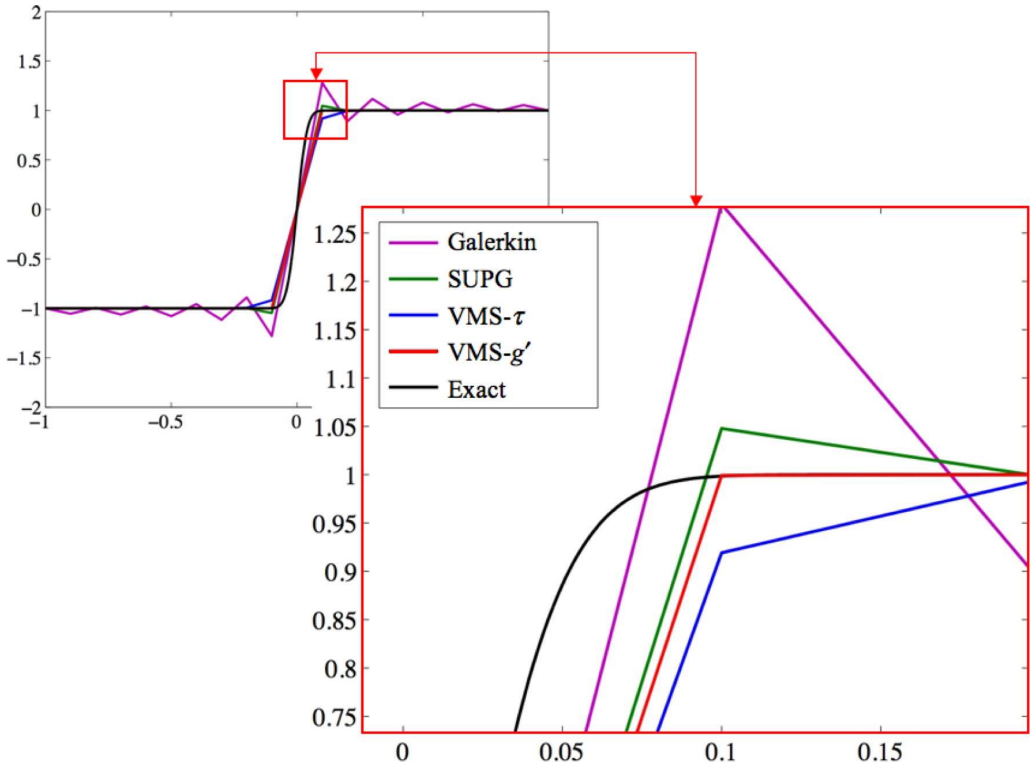


Figure 4.3: Results for advection-diffusion with non-constant velocity. The detail near the right edge of the layer highlights the differences between the methods.

Investigating this simple problem makes a couple of things apparent. The first is that VMS-τ seems to be more stable that SUPG. Though this lone example is insufficient to prove the point, it should not be overlooked. The second thing to note is that VMS-g' gives us exactly what we ask of it. In this case we wanted the H_1-optimal fit and that is what we obtained.

In an idealized setting, VMS-g' seems to be the best method, but this is rarely the case in practice. To use this method, we need to be able to find g'. In this simple setting we were able to solve a problem that was local to the element in order to find it, but in higher dimensions the problem will be truly global as the support of g' will be all of Ω. The single biggest problem is that in the vast majority of cases it will actually be impossible to analytically find g or g'. Moreover, in this example problem the g' of (4.42) required 300 *digits* of precision to evaluate it[5]. The implementation was performed in MATLAB where the precision of arithmetic can be specified, but solving the 20 element problem required more than five minutes because of the demands of such high precision arithmetic. Additionally, a very high quadrature rule had to be used because, despite the fact that

[5]If this function were to be used regularly, it is possible that a clever way to reorder the operations in its evaluation could be found that would decrease this demand for precision.

the elements are linear, g' is not a polynomial function.

In light of these difficulties in finding and using the fine-scale Green's operator, the developments of Hughes and Sangalli [39] as they stand are of more theoretical interest than practical applicability. It will be the goal of the remainder of this work to use the theoretical framework of [39] to find numerical approximations of g' and subsequently u' when analytical values are impossible to obtain.

Chapter 5

Numerical Modeling of the Fine Scales within the Variational Multiscale Method

In this chapter, we introduce a new approach to approximating the fine-scale entities that arise in the variational multiscale method. This will begin with a brief mention of the multiscale discontinuous Galerkin (MDG) method, which has served as a stepping stone for this work. We will then introduce our new approach in its full generality. Finally, numerous examples will be used to explore the features of the method.

5.1 The multiscale discontinuous Galerkin method

This section will introduce MDG and discuss it as it relates to our work, focusing on its multiscale representation of the numerical solution. In an effort to reconcile MDG with VMS we observe that, without explicitly intending to (or realizing it), MDG models the fine scales by approximating the *element* Green's function. Though the developments of Section 5.2 do not explicitly build on MDG, it has served as an inspiration in that it made us aware that the various fine-scale entities could be modeled numerically in such a way as to provide stability and robustness to a numerical method.

5.1.1 Overview of MDG

The discontinuous Galerkin (DG) method is felt to have advantages of robustness over the classical continuous Galerkin (CG) method, but this robustness comes at a price. The number of unknowns in a DG method can be many times that of its CG counterpart. For example, assuming about

seven linear tetrahedral elements per node, a DG system involves approximately 28 times as many degrees-of-freedom as the corresponding CG system [40]. The multiscale discontinuous Galerkin method [10, 17, 40] was developed in an attempt to combine the advantages of the continuous and discontinuous Galerkin methods. It is the aim of MDG to preserve the robustness of DG, without experiencing this "explosion" in the number of unknowns. It attempts to do so by *parameterizing* the discontinuous degrees-of-freedom by the continuous degrees-of-freedom.

Let us consider an abstract linear problem on domain Ω,

$$\mathcal{L}u = f \quad \text{in } \Omega, \tag{5.1}$$

with a given discontinuous Galerkin variational formulation: find $u^h \in V^h$ such that

$$B_{DG}(w^h, u^h) = L(w^h) \quad \forall w^h \in V^h, \tag{5.2}$$

where V^h is a specified finite-dimensional DG space. For example, $V^h = \{v^h \in L^2(\Omega) : v^h|_{\Omega_e} \in P^k(\Omega_e)\}$, where $P^k(\Omega_e)$ is the space of polynomials complete to degree k on element Ω_e. For this formulation to be valid, the bilinear form B_{DG} must include the appropriate flux terms and weak enforcement of boundary conditions typically associated with a DG method (see, *e.g.* [11] for a discussion of such mechanisms and their effects).

Rather than attempting to split the solution space of the original problem as in VMS, MDG formally splits the *discrete solution* into a continuous, "coarse" part and a discontinuous, "fine" part as

$$u^h = \bar{u}^h + u'^h. \tag{5.3}$$

The splitting of u^h is formal and it is the goal of MDG to define a problem that parameterizes u'^h by $\bar{u}^h$. The only assumption is that $\bar{u}^h \in V^h \bigcap C^0$ and that $u'^h \in V^h$. Note that, as it is the discrete solution u^h that we have split, we have a finite-dimensional basis for both our coarse- and fine-scale spaces.

After inserting (5.3) into (5.2), we could obtain coarse- and fine-scale equations analogous to the VMS approach of the previous chapter. As V^h is finite-dimensional, this would lead to a tractable problem for u'^h, but as it would have as many unknowns as our original DG problem, we would not have made any progress toward efficiency. Ideally, we would like to solve a local problem for the fine scales on each element rather than a large global problem, and indeed this might naively appear feasible as the supports of the discontinuous fine-scale weighting functions w'^h can be assumed local to individual elements, but it is not actually possible because the flux terms couple the local element problems together (we should not expect to be able to eliminate 27 out of 28 degrees-of-freedom and retain full accuracy). Some sort of approximation must be made

if we are to be left with only a local problem, and [10] mentions several such options for obtaining approximate, local, fine-scale problems.

The approach we take (not explicitly mentioned in [10]) is to simply write down a local discontinuous Galerkin problem for u' that has *weakly enforced* homogeneous Dirichlet boundary conditions and is driven by the residual of the coarse scales. We know from Chapter 4 and Equations (4.5) and (4.25) that the fine-scale problem is driven by the residual of the coarse scales. The homogeneous Dirichlet conditions that weakly enforce $u'^h|_{\partial\Omega_e} = 0$ are equivalent to weakly enforcing continuity of the total solution, that is, weakly enforcing $u^h|_{\partial\Omega_e} = \bar{u}^h$. The result is a formulation with all of the major features we require from our fine-scale problem, and it can be solved at the element level. The details will be presented for specific examples below, but abstractly we have the following local problem for $u'^h|_{\Omega_e}$:

$$B_e(w'^h, u'^h) = L_e(w'^h) - \bar{B}_e(w'^h, \bar{u}^h) \quad \forall w'^h \in V^h(\Omega_e), \tag{5.4}$$

where we have allowed the bilinear form containing $\bar{u}^h$ to have a different form than that for the fine scales alone. Note that the appropriate integrals over the element boundaries are included in B_e and $\bar{B}_e$.

At this point, let us introduce a basis on the element level for both our coarse- and fine-scale solutions. We need only include the functions with support on the current element (which is completely in keeping with the structure of standard finite element software). We have

$$\bar{u}^h(x) \;\; = \sum_{j=1}^{\bar{n}} \bar{c}_j \bar{N}_j(x) \;\; = \bar{\mathbf{U}} \cdot \bar{\mathbf{N}}, \tag{5.5}$$

$$u'^h(x) \;\; = \sum_{j=1}^{n'} d'_j N'_j(x) \;\; = \mathbf{U}' \cdot \mathbf{N}', \tag{5.6}$$

where $\bar{\mathbf{U}} \equiv [\bar{c}_1, \bar{c}_2, \ldots, \bar{c}_{\bar{n}}]^T$, $\mathbf{U}' \equiv [d'_1, d'_2, \ldots, d'_{n'}]^T$, $\bar{\mathbf{N}} \equiv [\bar{N}_1(x), \bar{N}_2(x), \ldots, \bar{N}_{\bar{n}}(x)]^T$, and $\mathbf{N}' \equiv [N'_1(x), N'_2(x), \ldots, N'_{n'}(x)]^T$.

Inserting (5.5) and (5.6) into (5.4) yields

$$\mathbf{K}'\mathbf{U}' = \mathbf{F} - \bar{\mathbf{K}}\bar{\mathbf{U}}, \tag{5.7}$$

where $\{\mathbf{K}'\}_{ij} = B_e(N'_i, N'_j)$, $\{\bar{\mathbf{K}}\}_{ij} = \bar{B}_e(N'_i, \bar{N}_j)$, and $\{\mathbf{F}\}_i = L_e(N'_i)$. Thus we can invert to obtain

$$\mathbf{U}' = \mathbf{G}\left(\mathbf{F} - \bar{\mathbf{K}}\bar{\mathbf{U}}\right), \tag{5.8}$$

where $\mathbf{G} = (\mathbf{K}')^{-1}$. The goal of MDG has been accomplished, at least at the element level. Equation (5.8) provides a direct relationship between the fine-scale degrees-of-freedom and the coarse-

scale degrees-of-freedom. Were we interested in pursuing MDG to its conclusion, we would use this local relationship to build the corresponding global relationship. Formally, the local quantities are never considered explicitly. Instead, all of this is then substituted back into (5.2) to obtain a global discontinuous Galerkin method with the number of degrees-of-freedom of a continuous method. Examples seem to indicate that many of the attractive stability properties of a standard DG method still hold true, see [40].

5.1.2 Connecting MDG and VMS

Rather than following the MDG methodology and returning to a global discontinuous Galerkin problem, let us investigate more thoroughly this representation of the fine scales. We may rewrite (5.7), summing on repeated indices, as

$$\begin{aligned} B_e(N_i', N_j')d_j' &= L_e(N_i') - \bar{B}_e(N_i', \bar{u}^h) \\ &= \left(N_i', f - \mathcal{L}\bar{u}^h\right)_{\Omega_e} \\ &= \int_{\Omega_e} N_i'(x) \left(f(x) - \mathcal{L}\bar{u}^h(x)\right) dx, \end{aligned} \tag{5.9}$$

where the second equality follows from integration-by-parts if the fine-scale problem has been formulated properly. Thus (5.8) may be rewritten as

$$d_j' = G_{ji} \int_{\Omega_e} N_i'(x) \left(f(x) - \mathcal{L}\bar{u}^h(x)\right) dx. \tag{5.10}$$

Now, recalling (5.6), we multiply (5.10) by $N_j'(y)$ and sum on repeated indices to obtain an expression for the fine-scale solution itself:

$$u'^h(y) = d_j' N_j'(y) = N_j'(y) G_{ji} \int_{\Omega_e} N_i'(x) \left(f(x) - \mathcal{L}\bar{u}^h(x)\right) dx. \tag{5.11}$$

Finally, we note that neither $N_j'(y)$ nor $\mathbf{G}$ are dependent upon the variable of integration x. Thus, let us define $g_h^e(x,y) \equiv N_j'(y) G_{ji} N_i'(x)$ and rewrite (5.11) as

$$u'^h(y) = \int_{\Omega_e} g_h^e(x,y) \left(f(x) - \mathcal{L}\bar{u}^h(x)\right) dx \quad \forall y \in \Omega_e. \tag{5.12}$$

Compare (5.12) with (4.31); they are form identical. The quantity $g_h^e(x,y)$ appearing implicitly (though never explicitly identified or used as such) in the MDG method is really an approxi-

mation to the element Green's function. The use of a basis to represent the fine scales, coupled with an appropriate local problem has provided not just an approximation to the fine-scale solution, but an approximation to some of the variational multiscale machinery that generates it. This observation motivates the remainder of this work.

5.2 Numerical modeling of the fine scales

Simply stated, it is the goal of this work to arrive at numerical approximations to the fine-scale entities arising in VMS that may be used to model the effect of the fine scales on the coarse scales. Though the above formulation arose within the context of a discontinuous Galerkin formulation of the global problem, it need not be so limited. Note that, in VMS, it is not u' that we are explicitly interested in, or even $\bar{u} + u'$, but simply $\bar{u}$. All along we have stated that $\bar{u}$ will be associated with the solution from our numerical method. This is not true of MDG. Recall that MDG does not split u into $\bar{u} = u^h$ and u', but rather splits u^h into $\bar{u}^h$ and u'^h. MDG requires a global DG formulation to solve for a discontinuous u^h. As we are interested in continuous u^h, we may consider inserting approximations similar in form to (5.12) anywhere that u' appears in our coarse-scale equation, even if that coarse-scale equation corresponds to a continuous method. In this way, we are never explicitly modeling u' but simply the effect of u' on $\bar{u} = u^h$.

5.2.1 Beyond the element Green's function

As we saw in Chapter 4, the element Green's function is not generally an object of particular interest in the variational multiscale method. In fact, the only context in which it is explicitly used is in 1D where H^1-optimality dictates that the fine-scale Green's function *is* the element Green's function, as was the case in Section 4.2.1. What we need is the global Green's function. With that in hand, we can use a linear projection of our choosing and the theory of Section 4.1.3 to obtain the object of real interest, the fine-scale Green's function.

The problem that we are confronted with is that the global Green's function is, by its very definition, globally supported (*i.e.*, it is supported on $\Omega \times \Omega$). We can consider posing a problem such as (5.4) over the entire domain and using it to generate a global version of (5.12), but this could be quite expensive to solve. What we would like is a local approximation to this global entity. The most obvious approach is to consider something in-between an element problem and a global problem. Toward this end, we will utilize local problems posed over a ***patch***[1] of elements: the union of several neighboring elements centered on the element in which we are currently assembling the coarse-scale problem.

[1] Do not confuse the present use of the word "patch" with the different usage of the word in the previous chapters on isogeometric analysis.

Different patch sizes will be considered, but typically each patch is much smaller than the total domain. The heuristic justification for this is two-fold. First, it was shown in [39] that for a point $x^* \in \Omega_e$, certain fine-scale Green's functions $g'(x^*, y)$ are strongly attenuated as y gets farther from x^*. Second, the use of weak boundary conditions allows the functions to be non-zero on the patch boundary, which is less restrictive than strong Dirichlet conditions on patch boundaries would be. Ideally, we would pose our fine-scale problem over the whole domain, but that not being possible, it stands to reason that the numerically computed patch Green's function g_h^p is a better approximation of the global Green's function g than is the element Green's function g_h^e.

We must also consider an appropriate basis for the fine scales. MDG used a discontinuous basis as it was inherited from the DG method. For our purposes, this is unnecessarily cumbersome, but we will retain one feature of that approach. Namely, we will represent all fine-scale entities using the basis for a space $V'^h \supset \bar{V}^h$, where $\bar{V}^h$ is our coarse-scale space. Though u'^h will be in some fine-scale space $\tilde{V}'^h$ such that $V'^h = \bar{V}^h \oplus \tilde{V}'^h$, it will be far simpler to work directly in V'^h and to utilize projections where appropriate to maintain the direct sum structure. In representing the fine scales, we always utilize a refined version of the unconstrained (*i.e.*, ignoring boundary conditions) coarse-scale space. Thus, we can characterize our fine-scale patch mesh by the following:

1. The coarse-scale element Ω_e that is currently being assembled.

2. The size of the patch, typically measured in the number of coarse-scale elements it includes. Figure 5.1 shows a 5×5 patch centered at Ω_e. At present we assume the patch size is constant except near the boundaries of the domain where it is truncated by $\partial\Omega$ in the obvious way.

3. The number of ***sub-elements*** each coarse-scale element is divided into via uniform h-refinement. The mesh in Figure 5.1 has a 3×3 sub-mesh, that is, each coarse element is divided uniformly into 9 smaller elements, three in each direction.

4. The polynomial order of the fine-scale basis, which is always greater than or equal to the order of the coarse basis.

With the patch and corresponding fine-scale mesh specified, we may pose our fine-scale problem: find $u'^h \in V'^h$ such that

$$B_{\Omega_e^P}(w'^h, u'^h) = L_{\Omega_e^P}(w'^h) - \bar{B}_{\Omega_e^P}(w'^h, \bar{u}^h), \quad \forall w'^h \in V'^h(\Omega_e^P) \tag{5.13}$$

where Ω_e^P is the patch corresponding to coarse element Ω_e, see Figure 5.1. This problem for the fine scales will have the major features discussed above. It will have weakly enforced homogeneous Dirichlet boundary conditions and it will be driven by the residual of the coarse scales.

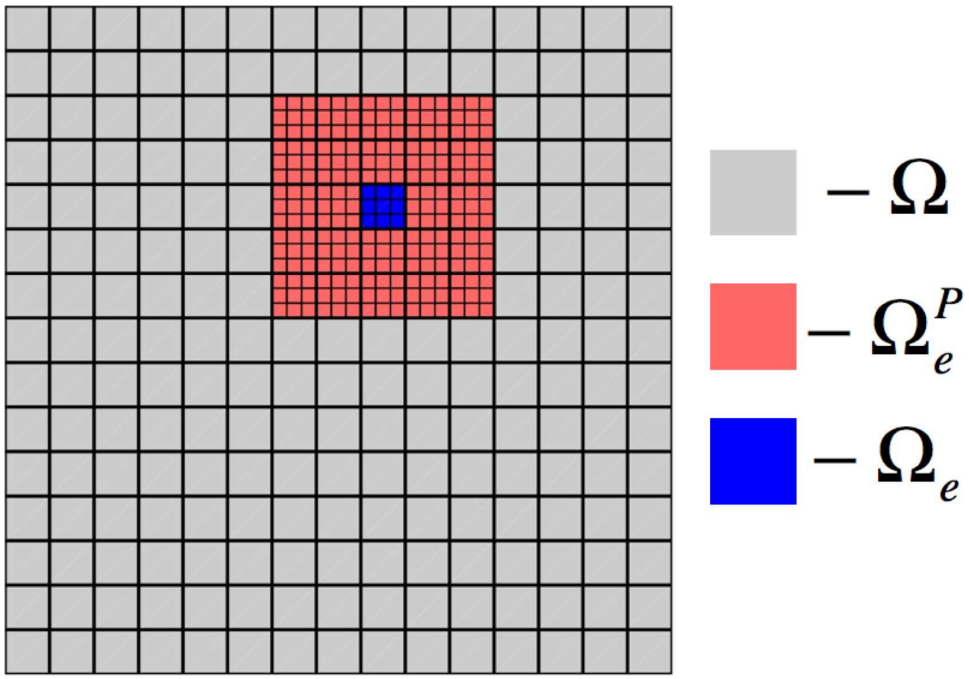


Figure 5.1: A 5×5 patch with a 3×3 sub-mesh of linear elements. The heavy lines are coarse-scale element boundaries. The thin lines are element boundaries of the refined mesh used to represent the fine scales. The polynomial order cannot be inferred from the picture, but we may assume it to be linear.

Remark 5.2.1. *To get a fine-scale problem, we always follow the same basic procedure. We begin by deriving the fine-scale problem that arises naturally through a variational multiscale treatment of the equation. This problem is then restricted to the patch and augmented with the weakly imposed boundary conditions as discussed. The fact that the fine-scale problems are driven by the residual of the coarse scales is inherited from VMS, not something we choose to impose.*

Using the bases of V'^h and $\bar{V}^h$ as in (5.5) and (5.6), from the formulation of (5.13) we can generate matrix $\mathbf{G}$:

$$\mathbf{G} = \left(\mathbf{K}'\right)^{-1}, \tag{5.14}$$

where $[\mathbf{K}']_{AB} = B_{\Omega_e^P}(N'_A, N'_B)$, which leads to the definition of our patch Green's function as

$$g_h^p(x,y) \equiv N'_i(y) G_{ij} N'_j(x) = \mathbf{N}'^T(y)\mathbf{G}\mathbf{N}'(x), \tag{5.15}$$

where $\mathbf{N}'(z) \equiv [N'_1(z), N'_2(z), \ldots, N'_{n'}(z)]^T$. *We consider our patch Green's function* $g_h^p(x,y)$ *to be a local approximation to the global Green's function, that is,* $g_h^p|_{\Omega_e \times \Omega_e} \approx g|_{\Omega_e \times \Omega_e}$.

5.2.2 From g_h^p to g_h'

Now that we have an approximation to the global Green's function in hand, we must select a projection in order to obtain an approximation to the fine-scale Green's function as in Section 4.1.3. To simplify notation, let us omit the superscript "h" and write V' and $\bar{V}$ rather than V'^h and $\bar{V}^h$, but we understand that these are both finite-dimensional spaces. We will define the projector $\mathcal{P} : V' \to \bar{V}$ through the use of an inner product $(\cdot,\cdot) : V' \times V' \to \mathbb{R}$. We have that for all $v \in V'$,

$$\bar{V} \ni \bar{v} = \mathcal{P}v \iff (\bar{w}, \bar{v}) = (\bar{w}, v) \quad \forall \bar{w} \in \bar{V}. \tag{5.16}$$

We can use (5.16) to construct an explicit representation for $\mathcal{P}$. First, let $\{\bar{N}_A\}_{A=1}^{\bar{n}}$ be a basis for $\bar{V}$ and $\{N'_A\}_{A=1}^{n'}$ be a basis for V'. Thus, let $\bar{w} = \bar{w}_A \bar{N}_A = \bar{\mathbf{W}}^T \bar{\mathbf{N}}$, where $\bar{\mathbf{W}} \equiv [\bar{w}_1, \bar{w}_2, \ldots, \bar{w}_{\bar{n}}]^T$ and $\bar{\mathbf{N}} \equiv [\bar{N}_1, \bar{N}_2, \ldots, \bar{N}_{\bar{n}}]^T$. As $\bar{V} \subset V'$ we can define an injection matrix $\mathbf{I}_{CF} : \mathbb{R}^{\bar{n}} \to \mathbb{R}^{n'}$ such that

$$\bar{w} = \bar{\mathbf{W}}^T \bar{\mathbf{N}} = \left(\mathbf{I}_{CF} \bar{\mathbf{W}}\right)^T \mathbf{N}' \quad \forall \bar{w} \in \bar{V}. \tag{5.17}$$

Now, let us elaborate on the terms in (5.16). We have

$$\begin{aligned} (\bar{w}, \bar{v}) &= \left(\bar{w}_A \bar{N}_A, \bar{v}_B \bar{N}_B\right) \\ &= \bar{w}_A \left(\bar{N}_A, \bar{N}_B\right) \bar{v}_B \\ &= \bar{\mathbf{W}}^T \mathbf{M}^C \bar{\mathbf{V}}, \end{aligned} \tag{5.18}$$

where $\left[\mathbf{M}^C\right]_{AB} = \left(\bar{N}_A, \bar{N}_B\right)$. Similarly,

$$\begin{aligned} (\bar{w}, v) &= \left(\bar{w}_A \bar{N}_A, v_B N'_B\right) \\ &= \left(\left[\mathbf{I}_{CF} \bar{\mathbf{W}}\right]_A N'_A, v_B N'_B\right) \\ &= \left[\mathbf{I}_{CF} \bar{\mathbf{W}}\right]_A \left(N'_A, N'_B\right) v_B \\ &= \bar{\mathbf{W}}^T \left(\mathbf{I}_{CF}\right)^T \mathbf{M}^F \mathbf{V}, \end{aligned} \tag{5.19}$$

where $\left[\mathbf{M}^F\right]_{AB} = (N'_A, N'_B)$. Putting things back together, we can use (5.18) and (5.19) to rewrite the right side (5.16) as

$$\begin{aligned} &\bar{\mathbf{W}}^T \mathbf{M}^C \bar{\mathbf{V}} = \bar{\mathbf{W}}^T \left(\mathbf{I}_{CF}\right)^T \mathbf{M}^F \mathbf{V} \quad \forall \bar{\mathbf{W}} \in \mathbb{R}^{\bar{n}} \\ &\Rightarrow \ \bar{\mathbf{V}} = \left(\mathbf{M}^C\right)^{-1} \left(\mathbf{I}_{CF}\right)^T \mathbf{M}^F \mathbf{V}, \end{aligned} \tag{5.20}$$

and thus we arrive at the following matrix representation $\mathbf{P}$ for operator $\mathcal{P}$:

$$\mathbf{P} = \left(\mathbf{M}^C\right)^{-1} \left(\mathbf{I}_{CF}\right)^T \mathbf{M}^F, \tag{5.21}$$

such that for all $v \in V'$,

$$\bar{v} = \mathcal{P}v \Longleftrightarrow \bar{\mathbf{V}} = \mathbf{P}\mathbf{V}, \tag{5.22}$$

where $v = \mathbf{V}^T\mathbf{N}'$ and $\bar{v} = \bar{\mathbf{V}}^T\bar{\mathbf{N}}$.

Note that the two projections used throughout the current work are the L^2-projection defined using the inner product

$$(w, v)_{L^2(\Omega_e^P)} = \int\limits_{\Omega_e^P} wv dV, \tag{5.23}$$

and the H^1-projection defined using

$$(w, v)_{H^1(\Omega_e^P)} = \int\limits_{\Omega_e^P} \boldsymbol{\nabla} w \cdot \boldsymbol{\nabla} v dV + \frac{1}{h} \int\limits_{\partial\Omega_e^P} wv dA, \tag{5.24}$$

where h is a length scale representative of the size of the patch.

Using (5.21) and (5.14), we can now numerically implement the theoretical construction of (4.15) to obtain an approximation to the fine-scale Green's function. Specifically,

$$g'_h(x, y) \equiv N'_i(y) G'_{ij} N'_j(x) = \mathbf{N}'^T(y)\mathbf{G}'\mathbf{N}'(x), \tag{5.25}$$

where

$$\mathbf{G}' = \mathbf{G} - \mathbf{G}\mathbf{P}^T \left(\mathbf{P}\mathbf{G}\mathbf{P}^T\right)^{-1} \mathbf{P}\mathbf{G}. \tag{5.26}$$

Not only can (5.25) be used to get an approximation of u' as

$$u'^h(y) = \int\limits_{\Omega_e} g'_h(x, y) \left(f(x) - \mathcal{L}\bar{u}^h(x)\right) dx \quad \forall y \in \Omega_e, \tag{5.27}$$

but we can also consider other approaches from the stabilized methods literature whose theoretical underpinnings involve a heretofore unobtainable fine-scale Green's function. For example we can now approximate residual-free bubbles as in [12]:

$$b_h(y) \equiv \int\limits_{\Omega_e} g'_h(x, y) dx \quad \forall y \in \Omega_e. \tag{5.28}$$

Similarly, we can now calculate rather than postulate appropriate stabilization parameters using

$$\tau_h \equiv \frac{1}{\text{meas}(\Omega_e)} \iint\limits_{\Omega_e \times \Omega_e} g_h'(x,y)dxdy. \tag{5.29}$$

The important thing to note is that each of these approaches are essentially parameter-free. All of the scaling information relating to element size and problem parameters are incorporated naturally through the fine-scale problem (5.13). Issues such as distorted elements, multiple dimensions, high polynomial orders, etc., are all treated within the same computational structure.

5.3 Advection-diffusion examples with numerically computed fine scales

We will apply the new method that we have developed to the advection-diffusion equation in several different settings. Basic entities will be examined in detail in one dimension. Two-dimensional examples will focus more on computational results.

Let us first derive a general VMS formulation for advection-diffusion, which will apply in all cases. Let Ω be an open, connected, bounded subset of $\mathbb{R}^d$, $d = 1, 2$, or 3, with piecewise smooth boundary $\Gamma = \partial\Omega$. Let $f : \Omega \to \mathbb{R}$ be a given source; $\mathbf{a} : \Omega \to \mathbb{R}^d$ is the velocity vector, assumed constant; $\mathbf{k} : \Omega \to \mathbb{R}^{d\times d}$ is the diffusivity tensor, assumed symmetric, positive-definite; and $g_b : \Gamma \to \mathbb{R}$ is the prescribed Dirichlet boundary data. The boundary value problem consists of solving the following equations for $u : \bar{\Omega} \to \mathbb{R}$ such that

$$\mathbf{a} \cdot \boldsymbol{\nabla} u - \boldsymbol{\nabla}(\mathbf{k}\boldsymbol{\nabla} u) = f \quad \text{in } \Omega, \tag{5.30}$$

$$u = g_b \quad \text{on } \partial\Omega, \tag{5.31}$$

where we interpret $\boldsymbol{\nabla}$ to be the gradient operator appropriate to the spatial dimension being considered. Let us define the solution and weighting spaces as

$$V = H^1_{g_b}(\Omega) = \{u | u \in H^1(\Omega),\ u = g_b \text{ on } \Gamma\}, \tag{5.32}$$

$$V_0 = H^1_0(\Omega) = \{u | u \in H^1(\Omega),\ u = \text{ on } \Gamma\}, \tag{5.33}$$

respectively. Thus the variational counterpart of (5.30) is: find $u \in \mathcal{V}$ such that

$$(-\boldsymbol{\nabla} w, \mathbf{a}u - \mathbf{k}\boldsymbol{\nabla} u)_\Omega = (w, f)_\Omega \quad \forall w \in \mathcal{V}_0, \tag{5.34}$$

where $(\cdot,\cdot)_\Omega$ denotes the L^2-inner product on Ω.

We will now consider a multiscale splitting of the solution and weighting functions as in

(4.3). With $u = \bar{u} + u'$, let us insert weighting function $\bar{w} \in \bar{V}_0 \subset H_0^1(\Omega)$ into (5.34) to get our coarse-scale equation: find $\bar{u} \in \bar{V}$ such that

$$(-\boldsymbol{\nabla}\bar{w}, \mathbf{a}\bar{u} - \mathbf{k}\boldsymbol{\nabla}\bar{u})_\Omega + \left(\mathcal{L}^*\bar{w}, u'\right)_\Omega = (\bar{w}, f)_\Omega \quad \forall \bar{w} \in \bar{V}, \tag{5.35}$$

where $\mathcal{L}^*(\cdot) = -\mathbf{a} \cdot \boldsymbol{\nabla}(\cdot) - \boldsymbol{\nabla}(\mathbf{k}\boldsymbol{\nabla}(\cdot))$.

We now turn our attention to the fine-scale problem. As in Section 5.2.1, for each element in the mesh we will pose a local problem over a patch which contains it. As stated, this problem for the fine scales is driven by the residual of the coarse scales and has homogeneous boundary conditions on u'. We seek $u' \in V'$ such that $\forall w' \in V_0'$

$$\begin{aligned}
&\left(-\boldsymbol{\nabla}w', \mathbf{a}u' - \mathbf{k}\boldsymbol{\nabla}u'\right)_{\Omega_e^P} \\
&- \left(\mathcal{L}^*w'\tau, \mathcal{L}u' - R'(\bar{u})\right)_{\tilde{\Omega}_e^P} \\
&+ \left(\chi_{out}\mathbf{a} \cdot \mathbf{n}w', u'\right)_{\Gamma_e^P} \\
&- \left(w', \mathbf{k}\boldsymbol{\nabla}u' \cdot \mathbf{n}\right)_{\Gamma_e^P} \\
&+ \left(s\mathbf{k}\boldsymbol{\nabla}w' \cdot \mathbf{n}, u'\right)_{\Gamma_e^P} \\
&+ \left(\frac{\epsilon|\mathbf{k}|}{h_\perp}w', u'\right)_{\Gamma_e^P} \\
&= \left(w', R'(\bar{u})\right)_{\Omega_e^P},
\end{aligned} \tag{5.36}$$

where $\mathcal{L}(\cdot) = \mathbf{a} \cdot \boldsymbol{\nabla}(\cdot) - \boldsymbol{\nabla} \cdot (\mathbf{k}\boldsymbol{\nabla}(\cdot))$ is the linear operator, $\mathcal{L}^*(\cdot) = -\mathbf{a} \cdot \boldsymbol{\nabla}(\cdot) - \boldsymbol{\nabla} \cdot (\mathbf{k}\boldsymbol{\nabla}(\cdot))$ is the adjoint of the operator, $R'(\bar{u})$ is the residual of the coarse scales projected into the fine scales as given by

$$\left(w', R'(\bar{u})\right)_{\Omega_e^p} = \left(w', f - \mathcal{L}\bar{u}\right)_{\Omega_e^p} \quad \forall w' \in V', \tag{5.37}$$

Γ_e^P is the boundary of patch Ω_e^p, $\mathbf{n} = \{n_i\}_{i=1}^d$ is the unit normal to the boundary, χ_{out} is the outflow characteristic function

$$\chi_{out} = \begin{cases} 1 & \mathbf{a} \cdot \mathbf{n} > 0, \\ 0 & \text{otherwise}, \end{cases} \tag{5.38}$$

and $h_\perp$ is a length representative of the size of the element in the direction orthogonal to the current edge, defined by

$$h_\perp = 2n_i \frac{\partial x_i}{\partial \xi_\alpha}\hat{n}_\alpha, \tag{5.39}$$

where $\hat{n}_\alpha = \tilde{n}_\alpha / \|\tilde{n}_\alpha\|$ and

$$\tilde{n}_\alpha = n_j \frac{\partial x_j}{\partial \xi_\alpha}. \tag{5.40}$$

It is easiest to understand (5.36) by examining the Euler-Lagrange equations in terms of the total solution $u = \bar{u} + u'$. Assuming for the moment that $\tau = 0$, we have

$$\int_{\Omega_e^P} w'(\mathbf{a}\cdot\boldsymbol{\nabla}u - \boldsymbol{\nabla}\cdot(\mathbf{k}\boldsymbol{\nabla}u) - f) + \epsilon\int_{\Gamma_e^P}\frac{|\mathbf{k}|}{h_\perp}w'(u-\bar{u}) + \int_{\Gamma_e^P}\chi_{in}w'\mathbf{a}\cdot\mathbf{n}(\bar{u}-u) - s\int_{\Gamma_e^P}\mathbf{k}\boldsymbol{\nabla}w'\cdot\mathbf{n}(\bar{u}-u) = 0, \tag{5.41}$$

where $\chi_{in} = 1 - \chi_{out}$ is the inflow characteristic function. We have arrived at (5.36) by using the formulation of Bazilevs and Hughes [7] to *weakly impose continuity of the total solution*, that is $u = \bar{u}$ on Γ_e^P.

The VMS-τ stabilization term on the second line of (5.36) has been added to account for some oscillations in the fine-scale entities that will be apparent in some of the examples. As we have assumed a finite-dimensional basis for the fine-scale space, we still have *unresolved* scales missing and there are cases where modeling their effect in the fine-scale equation is useful. However, we will frequently not need this term and set $\tau = 0$. Note that all of the approximation has been pushed onto the fine-scale problem, while (5.35) remains an exact equation for $\bar{u}$. The selection of s and ϵ will be discussed in the examples as well.

Defining

$$\begin{aligned} B_{\Omega_e^P}(w',u') = &\left(-\boldsymbol{\nabla}w', \mathbf{a}u' - \mathbf{k}\boldsymbol{\nabla}u'\right)_{\Omega_e^P} - \left(\mathcal{L}^*w'\tau, \mathcal{L}u'\right)_{\tilde{\Omega}_e^P} \\ &+ \left(\chi_{out}\mathbf{a}\cdot\mathbf{n}w', u'\right)_{\Gamma_e^P} - \left(w', \mathbf{k}\boldsymbol{\nabla}u'\cdot\mathbf{n}\right)_{\Gamma_e^P} \\ &+ \left(s\mathbf{k}\boldsymbol{\nabla}w'\cdot\mathbf{n}, u'\right)_{\Gamma_e^P} + \left(\frac{\epsilon|\mathbf{k}|}{h_\perp}w', u'\right)_{\Gamma_e^P}, \end{aligned} \tag{5.42}$$

$$L_{\Omega_e^P}(w') = \left(w', f\right)_{\Omega_e^p}, \tag{5.43}$$

and

$$\bar{B}_{\Omega_e^P}(w',\bar{u}) = \left(w', \mathcal{L}\bar{u}\right)_{\Omega_e^p} - \left(\mathcal{L}^*w'\tau, R'(\bar{u})\right)_{\tilde{\Omega}_e^P}, \tag{5.44}$$

lets us rewrite (5.36) in the form of (5.13). Thus we have a specific definition for our matrix

$$\left[\mathbf{K}'\right]_{AB} = B_{\Omega_e^P}(N_A', N_B'). \tag{5.45}$$

5.3.1 1D examples

Let us begin with the simplest of all possible cases. We will use piecewise linear functions as a basis to the coarse-scale space and assume the $\mathcal{P}$ is the H^1-projector. In this way, we can work directly

with the element Green's function and avoid explicitly building a projector. In this special case, we pose (5.36) on the element and (5.45), (5.14), and (5.15) lead directly to $g'_h = g^e_h$. This simple setting will also allow us to compare with analytically computed values for many of the objects of interest.

To proceed, all we need is to select a basis for our fine scales. Again, let us begin with simple piecewise linear functions with no sub-mesh (*i.e.*, at the element level we are using the same basis for the fine scales as for the coarse scales). Setting $\tau = 0$, we can solve for $g'_h : (0,h) \times (0,h) \to \mathbb{R}$ in terms of parameters s and ϵ, as well as the advective velocity a, the element size h, and the grid Peclet number $\alpha \equiv \frac{ah}{2\kappa}$. Additionally, using (5.28) and (5.29) we can solve for $b_h : (0,h) \to \mathbb{R}$ and $\tau_h \in \mathbb{R}$ as well. Namely,

$$g'_h(x,y) = \frac{2\alpha(2y(\epsilon+\alpha) - \epsilon h)x - 2\alpha(y(2\alpha h + \epsilon h) - \alpha h^2 - sh^2)}{ah^2(2\alpha^2 + 2\alpha(\epsilon+s) + \epsilon^2 + 2\epsilon s)} \tag{5.46}$$

$$b_h(y) = \frac{\alpha(h\epsilon + 2\alpha y + 2hs)}{a(2\alpha^2 + 2\alpha(\epsilon+s) + \epsilon^2 + 2\epsilon s)} \tag{5.47}$$

$$\tau_h = \frac{h}{2a}\left(\frac{2\alpha(\alpha+\epsilon+2s)}{2\alpha^2 + 2\alpha(\epsilon+s) + \epsilon^2 + 2\epsilon s}\right) \tag{5.48}$$

We need to choose values for s and ϵ. We can get some insight by comparing these results with the analytically computed entities. The analytical stabilization parameter in this context is $\tau = \frac{h}{2a}(\coth(\alpha) - 1/\alpha)$. Looking at τ_h in the advection-dominated limit we have

$$\lim_{\alpha\to\infty} \tau_h = \frac{h}{2a}, \tag{5.49}$$

which matches the analytical result exactly. Already, such a crude discretization for the fine scales is yielding promising results. For the diffusion-dominated case we get

$$\lim_{\alpha\to 0} \tau_h = 0, \tag{5.50}$$

as we should, but we can learn more by taking the derivative with respect to the grid Peclet number to see the slope with which the diffusive limit is approached:

$$\lim_{\alpha\to 0} \frac{d\tau_h}{d\alpha} = \left(\frac{h}{2a}\right)\left(\frac{2}{\epsilon}\right). \tag{5.51}$$

The analytical result is matched identically for $\epsilon = 6$, and so that is what we will choose. As s is as of yet unrestricted, we will select $s = 1$, leading to a skew-symmetric method.

Turning now to the fine-scale Green's function, we need an expression to compare our re-

sults with. The analytical expression for the element Green's function (see, *e.g.*, [31, 33, 39]) is

$$g^e(x,y) = \begin{cases} \dfrac{\left(1-e^{-\frac{ah}{\kappa}\left(1-\frac{y}{h}\right)}\right)\left(1-e^{-\frac{ax}{\kappa}}\right)}{a\left(1-e^{-\frac{ah}{\kappa}}\right)} & \text{if } x \le y, \\ \dfrac{\left(e^{\frac{ay}{\kappa}}-1\right)\left(e^{-\frac{ax}{\kappa}}-e^{-\frac{ah}{\kappa}}\right)}{a\left(1-e^{-\frac{ah}{\kappa}}\right)} & \text{if } x \ge y. \end{cases} \tag{5.52}$$

Again, in our current context of H^1-optimality and linear coarse-scale elements we have $g' = g^e$. Figure 5.2 shows g' and our one-linear-element approximation, g'_h, for the extremely advection-dominated[2] case of $\alpha = 10^6$. Note that the boundary condition is being ignored. This is essential, as enforcing a homogeneous boundary condition on a linear element would result in $g'_h \equiv 0$. Interestingly, the inclusion of the boundary terms in (5.36) is necessary for ensuring that our $\mathbf{K}'$ matrix is invertible. Given how thin the boundary layers in g' are, it is easy to heuristically argue that this is as good of an approximation as our basis is capable of.

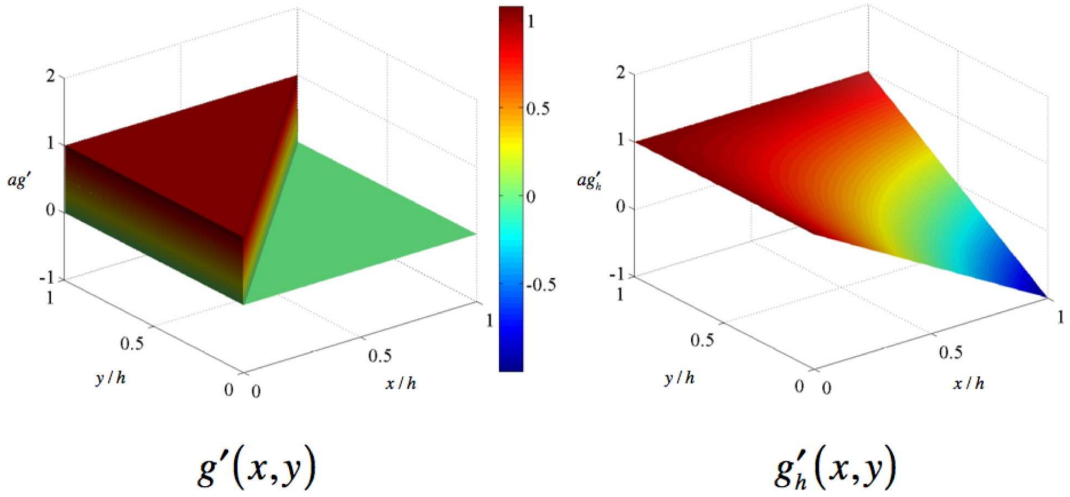


Figure 5.2: Exact, g', and approximate, g'_h, fine-scale Green's functions for the advection-dominated case of $\alpha = 10^6$. One bilinear element is used to model g'_h. The coarse scales are also linear.

The surprising quality of the approximation is even more striking when we consider the exact and approximate bubbles b and b_h, respectively. They are shown, again for $\alpha = 10^6$, in Figure 5.3. The infinitesimally thin layer at the right-most edge of b is completely ignored in b_h. As with g'_h, this ability to ignore boundary layers that the basis is incapable of representing is crucial to the success of the method in accurately approximating the function in the rest of the element.

When we consider a diffusion-dominated case, $\alpha = 10^{-1}$, results suffer for such a coarse

[2]We are exploring the advection- and diffusion-dominated limits by keeping $a = 1$ fixed and varying κ.

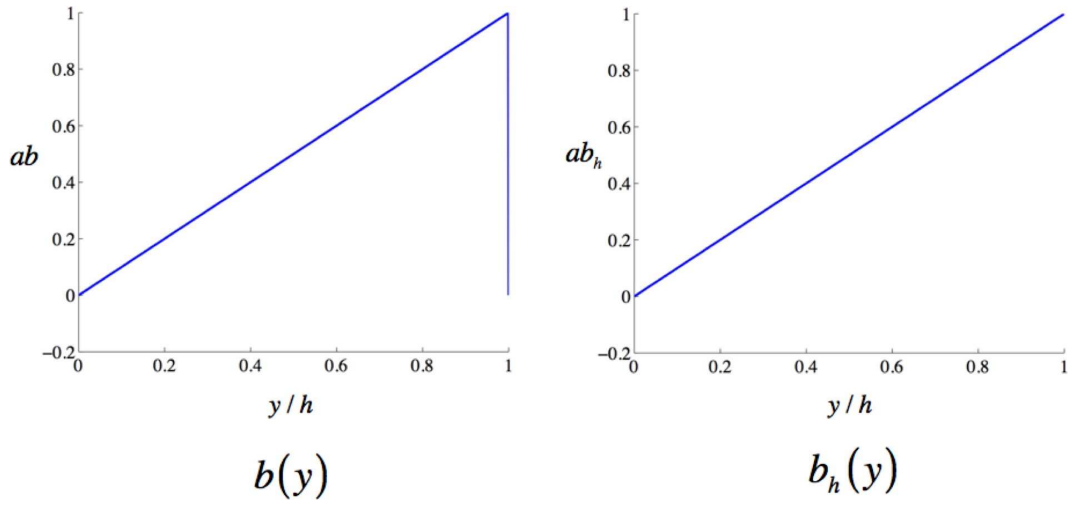


Figure 5.3: Exact, b, and approximate, b_h, residual-free bubbles for the advection-dominated case of $\alpha = 10^6$. One linear element is used to model b_h. The coarse scales are also linear.

basis. Though this is disappointing, it should be noted that the diffusion-dominated case is not the "hard case," that is, it is not the case in which stability of the coarse-scale equation suffers. From Figures 5.4 and 5.5, showing the fine-scale Green's functions and residual-free bubbles, respectively, we see that the main reason for the degradation in quality is that the objects we are approximating are simply very far from linears and thus difficult for our basis to represent. Still, the mean values are fairly well represented for all values of α, as we see by examining τ_h in Figure 5.6.

As we enrich the fine-scale space from one linear element to one quadratic element (leaving the linear coarse-scale space untouched), we do see a gain in accuracy. Explicit expressions for g_h' and b_h can be obtained using a symbolic mathematics package, but they are so long and unwieldy that we see no need for including them here. These entities can be explored quite simply through purely numerical means. Still, it is informative to examine the stabilization parameter:

$$\tau_h = \frac{h}{2a}\left(\frac{\alpha(2\alpha^2 + (16s\epsilon + 2\epsilon)\alpha + 12s^2 + \epsilon^2)}{2\alpha^3 + (6s + 2\epsilon)\alpha^2 + (6\epsilon + 24s)\alpha + 36s^2 + 3\epsilon^2 + 24s\epsilon}\right). \tag{5.53}$$

As with linears, this τ_h has the correct advective and diffusive limit values, but with this enriched basis we now also get the correct slope in the diffusion-dominated limit as well. That is, for any choice of ϵ we have

$$\lim_{\alpha\to 0}\frac{d\tau_h}{d\alpha} = \left(\frac{h}{2a}\right)\left(\frac{1}{3}\right). \tag{5.54}$$

Thus by enriching the basis, more of the actual dynamics of the situation are captured without requiring an expert user to tune parameters. We leave $s = 1$ as before, but now select $\epsilon = 0$ simply

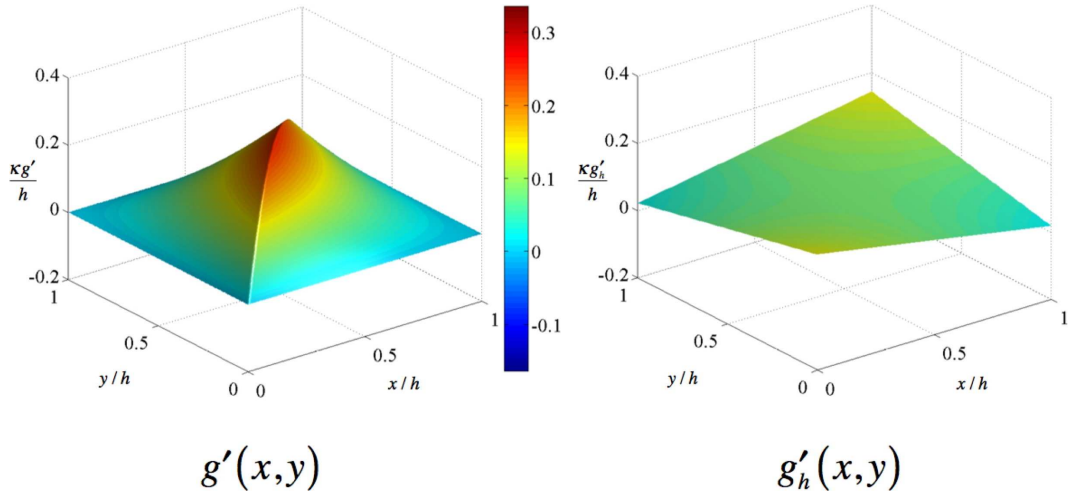


Figure 5.4: Exact, g', and approximate, g'_h, fine-scale Green's functions for the diffusion-dominated case of $\alpha = 10^{-1}$. One bilinear element is used to model g'_h. The coarse scales are also linear.

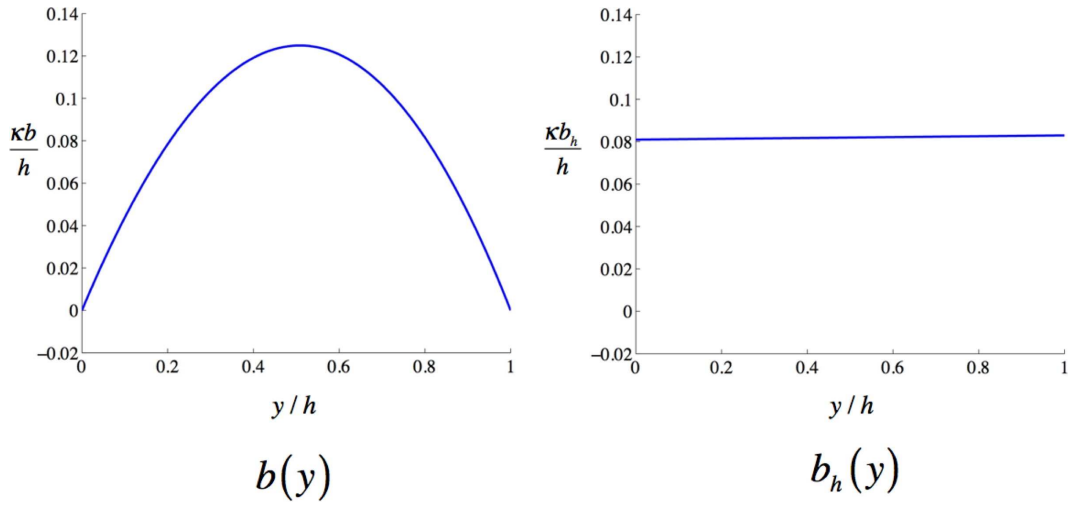


Figure 5.5: Exact, b, and approximate, b_h, residual-free bubbles for the diffusion-dominated case of $\alpha = 10^{-1}$. One linear element is used to model b_h. The coarse scales are also linear.

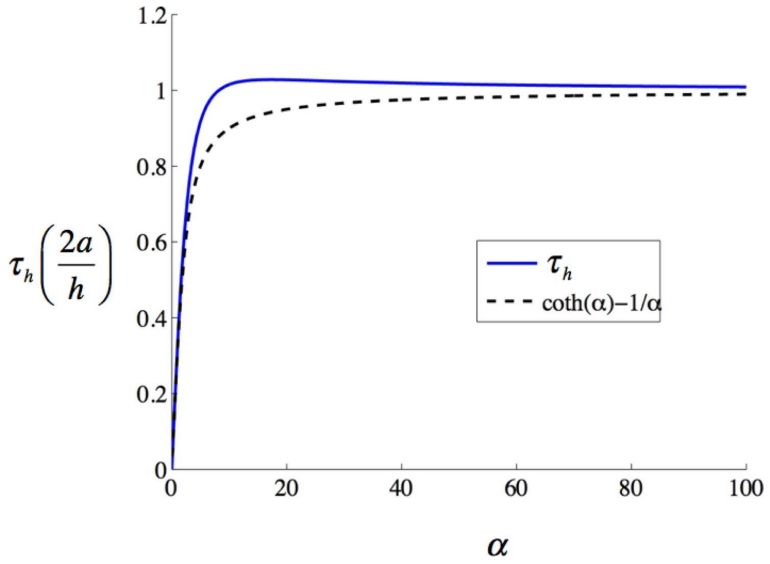


Figure 5.6: τ_h as a function α. The analytical value is given by the dotted line. Our numerical result is in blue. Note that they do converge for large α as shown in (5.49).

to remove it as a consideration (setting it equal to zero is as close as we can get to not having to consider it at all).

Figure 5.7 shows our biquadratic representation g'_h for our advection-dominated case of $\alpha = 10^6$, again compared with the exact value given by (5.52). As before, the boundary conditions are being seemingly ignored, to the benefit of the result. This single element approximation shows even more of the character of the exact solution than we saw in Figure 5.2. The result for b_h in Figure 5.8 is virtually identical to that of Figure 5.3, which is to say that it is indistinguishable from the exact result in all but the vanishingly thin boundary layer of the exact solution.

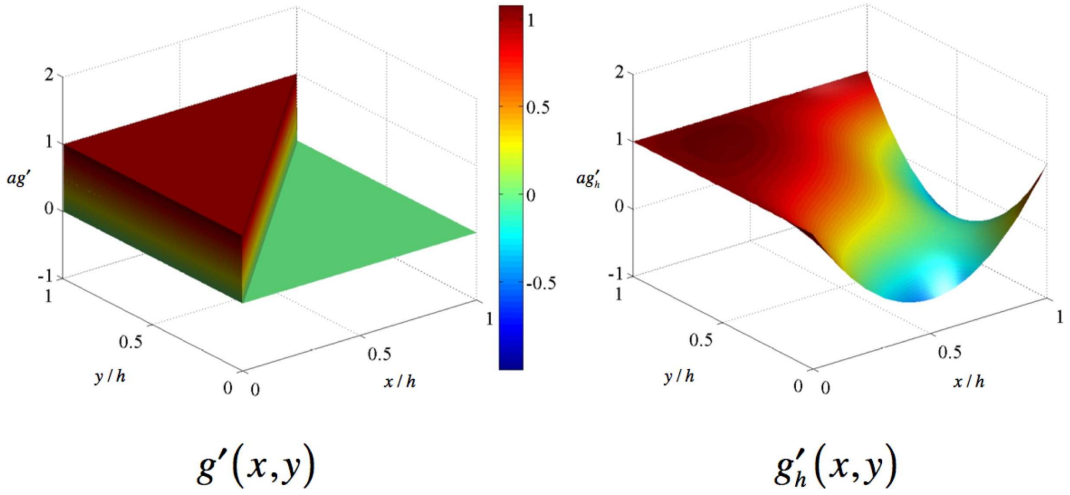


Figure 5.7: Exact, g', and approximate, g'_h, fine-scale Green's functions for the advection-dominated case of $\alpha = 10^6$. One biquadratic element is used to model g'_h. The coarse scales are linear.

Considering now the diffusive case of $\alpha = 10^{-1}$, it is not immediately evident from the fine-scale Green's function that we have improved over our previous linear approximation, see Figure 5.9. When we consider the residual-free bubble, however, it is clear that we are indeed benefitting from the enriched basis. Figure 5.10 shows that our quadratic bubble is far more accurate than was the linear bubble of Figure 5.5. As before, the τ_h is well-behaved for all values of α, see Figure 5.11.

We can enrich the basis further by continuing on to higher polynomial orders. Figure 5.12 shows the result for cubic and quartic single-element approximations in the advection-dominated case. Our other option for enrichment is to introduce a sub-mesh and use an h-refined fine-scale mesh of lower order elements. Figure 5.13a shows the result of using multiple linear elements without stabilizing the fine-scale problem. There is a node-to-node oscillation present that we assume is due to the unaccounted for effect of the unresolved scales. When we stabilize the problem by

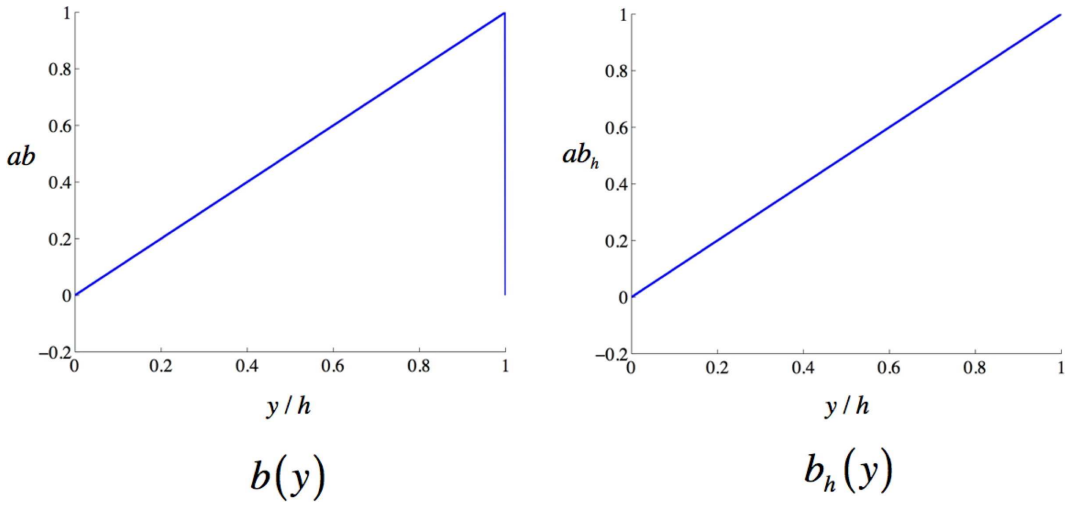


Figure 5.8: Exact, b, and approximate, b_h, residual-free bubbles for the advection-dominated case of $\alpha = 10^6$. One quadratic element is used to model b_h. The coarse scales are linear.

turning on the VMS-τ term using the classical τ in (5.36), we get the results shown in Figure 5.13b.

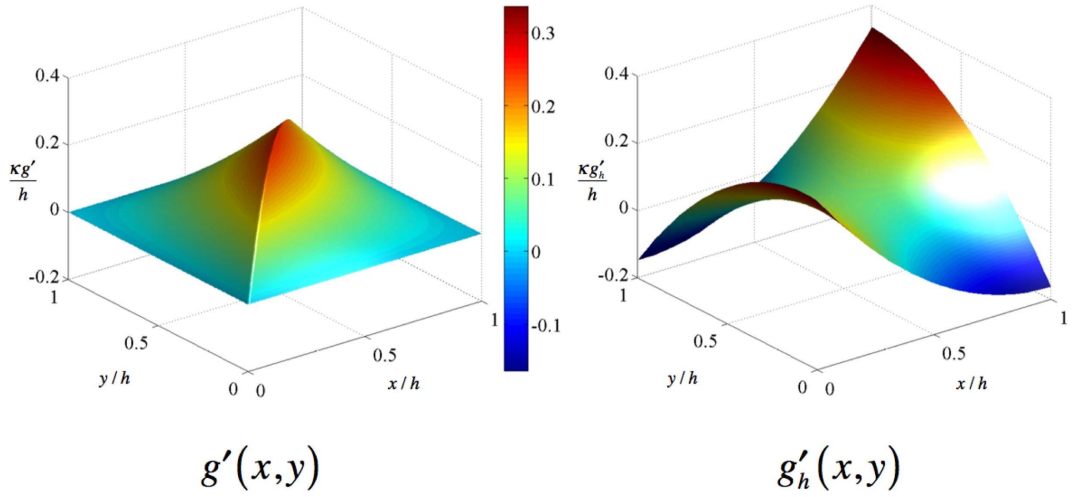


Figure 5.9: Exact, g', and approximate, g'_h, fine-scale Green's functions for the diffusion-dominated case of $\alpha = 10^{-1}$. One biquadratic element is used to model g'_h. The coarse scales are linear.

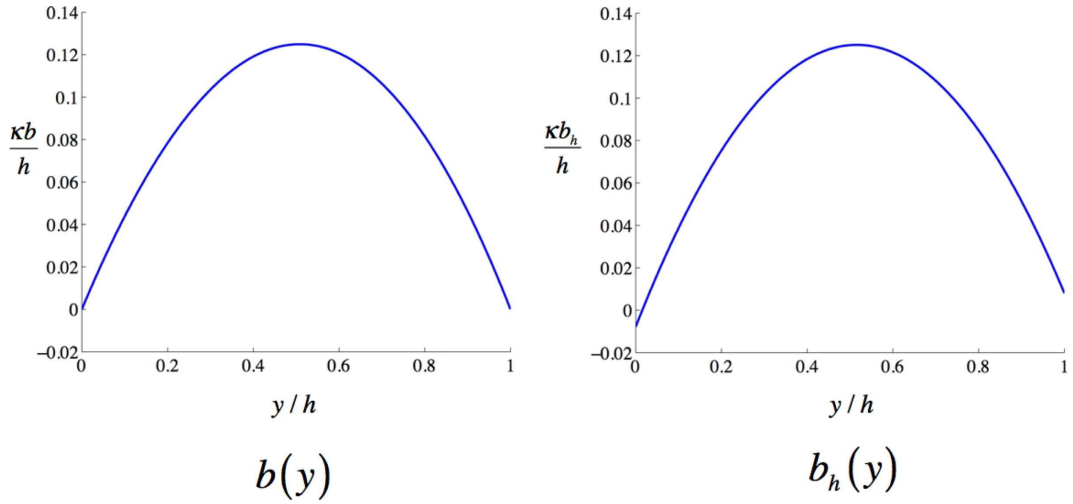


Figure 5.10: Exact, b, and approximate, b_h, residual-free bubbles for the diffusion-dominated case of $\alpha = 10^{-1}$. One quadratic element is used to model b_h. The coarse scales are linear.

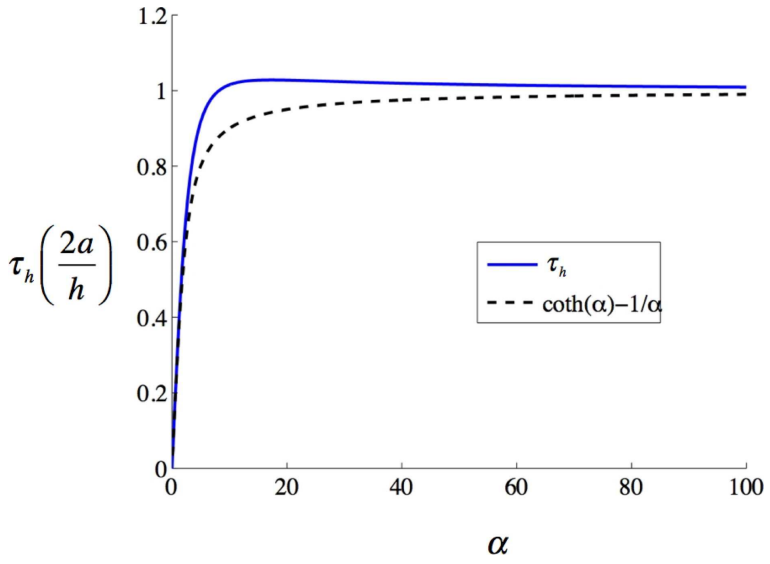


Figure 5.11: τ_h as a function α. The analytical value is given by the dotted line. Our numerical result is in blue.

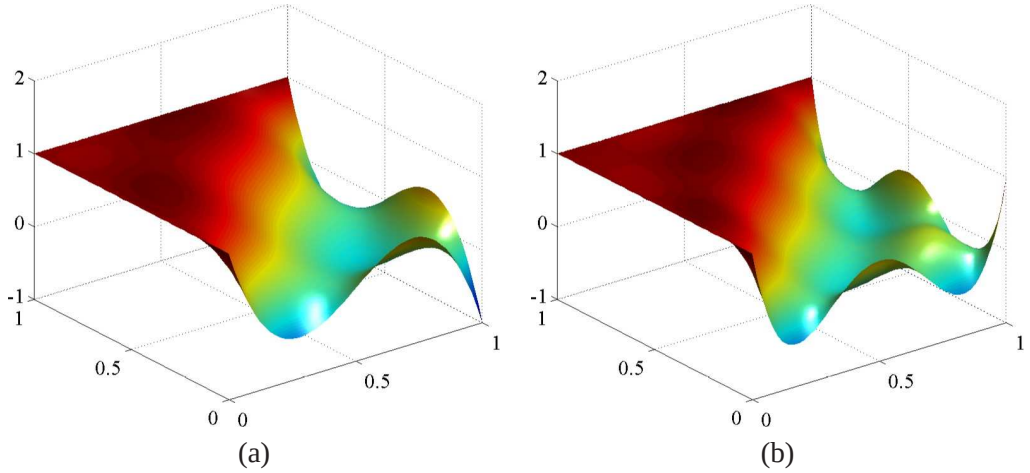


Figure 5.12: Higher-order single-element approximations to the fine-scale Green's function with $\alpha = 10^6$. a) Cubic basis functions. b) Quartic basis functions.

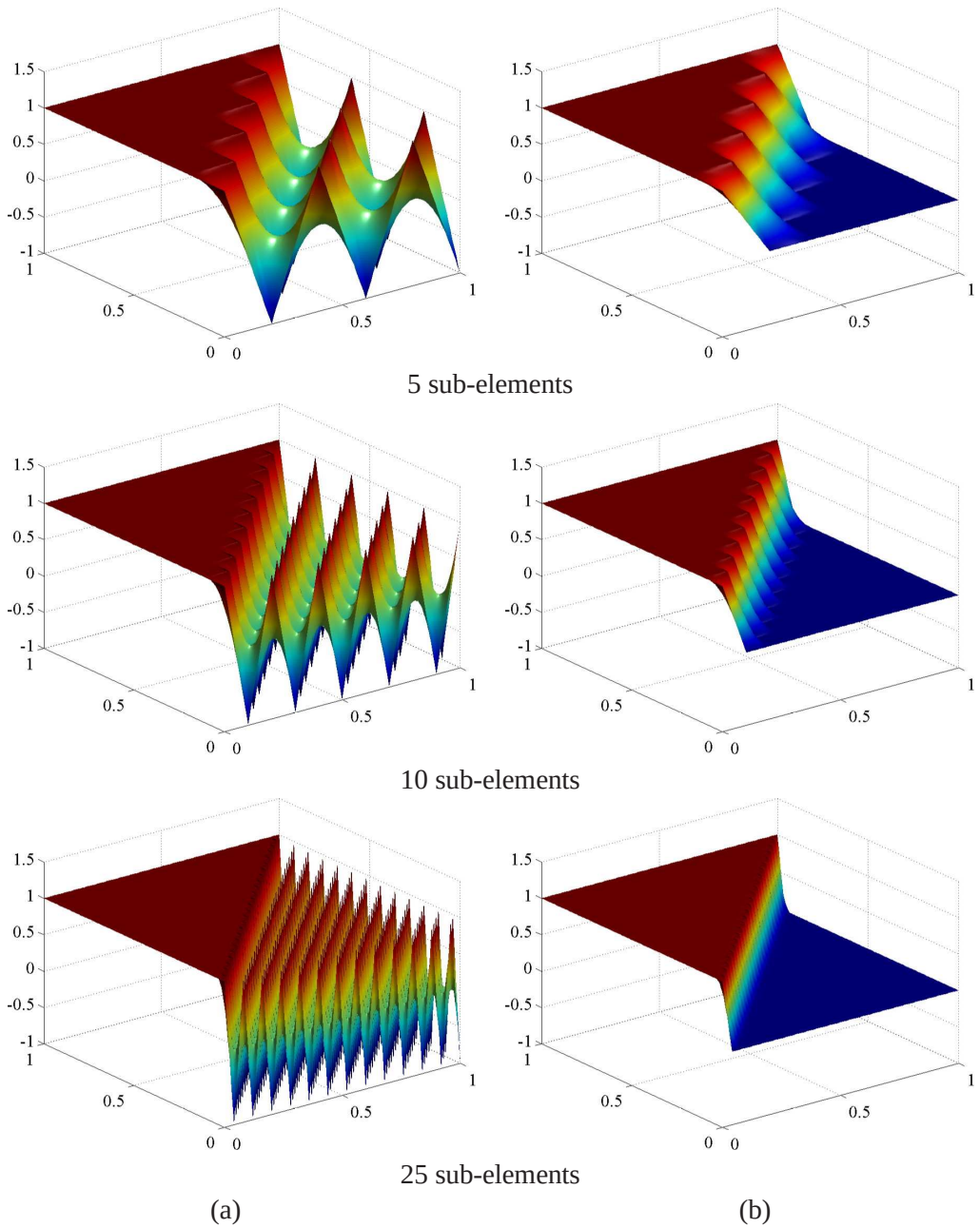


Figure 5.13: Linear approximations to the fine-scale Green's function with $\alpha = 10^6$. The coarse scales are also linear. a) Unstabilized approximations. b) Stabilized approximations.

If we now consider the case of higher-order coarse scales, the relationship $g' = g^e$ no longer holds true. If we continue to seek H^1-optimality, the end-nodes of the coarse-scale element will still interpolate the exact solution, but there are now polynomial bubbles in the coarse-space which must be removed from the fine-scale space. In [39] this is accomplished analytically for the 1D case to arrive at an expression which holds true for a coarse-scale basis of polynomial order k:

$$\begin{aligned} g'(x,y) =& g^e(x,y) - \left[\int_0^h g^e(\tilde{x},y)d\tilde{x} \quad \dots \quad \int_0^h \tilde{x}^{k-2} g^e(\tilde{x},y)d\tilde{x}\right] \\ & \times \begin{bmatrix} \int_0^h \int_0^h g^e(\tilde{x},\tilde{y})d\tilde{x}d\tilde{y} & \dots & \int_0^h \int_0^h \tilde{x}^{k-2} g^e(\tilde{x},\tilde{y})d\tilde{x}d\tilde{y} \\ \vdots & \ddots & \vdots \\ \int_0^h \int_0^h \tilde{y}^{k-2} g^e(\tilde{x},\tilde{y})d\tilde{x}d\tilde{y} & \dots & \int_0^h \int_0^h \tilde{x}^{k-2}\tilde{y}^{k-2} g^e(\tilde{x},\tilde{y})d\tilde{x}d\tilde{y} \end{bmatrix}^{-1} \\ & \times \begin{bmatrix} \int_0^h g^e(x,\tilde{y})d\tilde{y} \\ \vdots \\ \int_0^h \tilde{y}^{k-2} g^e(x,\tilde{y})d\tilde{y} \end{bmatrix}. \end{aligned} \tag{5.55}$$

For quadratic coarse-scale elements, let us approximate the fine scales with a single quadratic element. We begin by approximating the element Green's function with g_h^e (shown in Figure 5.7, though in that case we identified it as g_h' as we were considering a linear coarse-scale basis). Now we insert this into (5.55), with $k = 2$ and replacing g^e with g_h^e everywhere that it appears, to get an expression g_h'. The result is shown in Figure 5.14, compared with the analytical result. As in the linear case, the basic character of the function being approximated is evident, though the thin boundary layer is being ignored. Figure 5.15 shows unstabilized and stabilized approximations using an h-refined sub-mesh for the fine scales.

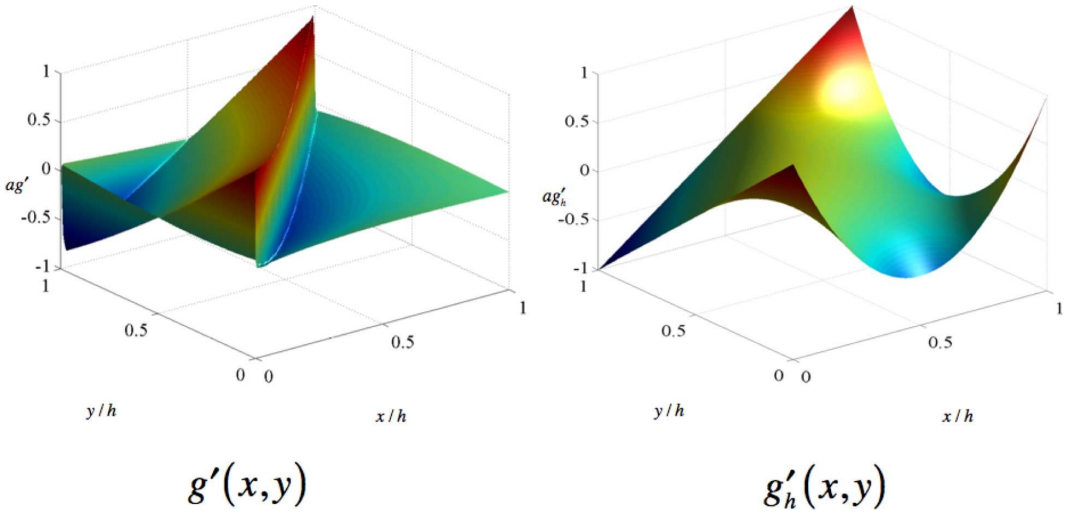


Figure 5.14: Exact, g', and approximate, g'_h, fine-scale Green's functions for the advection-dominated case of $\alpha = 10^6$. One biquadratic element is used to model g'_h. The coarse scales are quadratic.

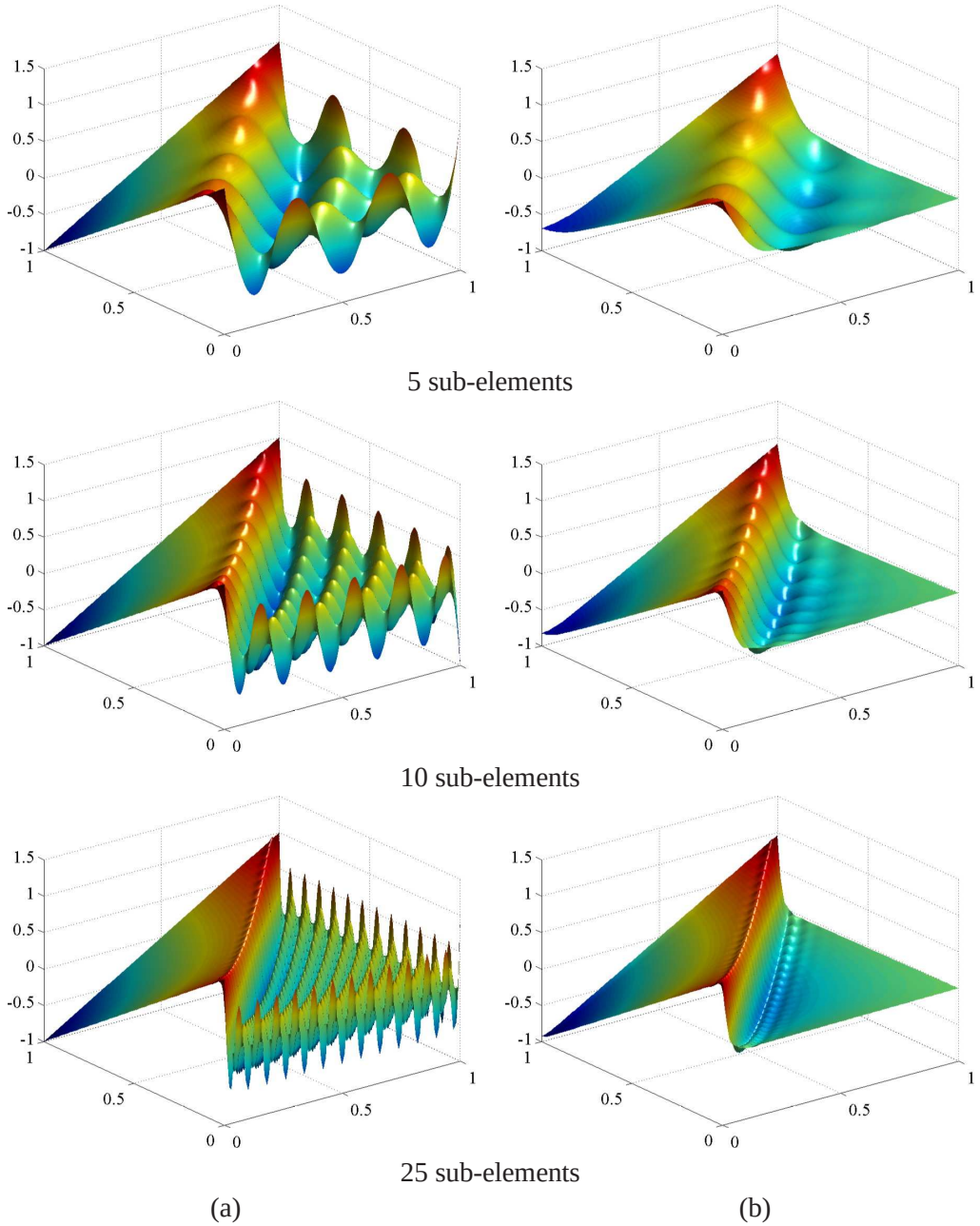


Figure 5.15: Quadratic approximations to the fine-scale Green's function with $\alpha = 10^6$. The coarse scales are also quadratic. a) Unstabilized approximations. b) Stabilized approximations.

Though plotting our fine-scale Green's functions is helpful in gaining intuition about the nature of the approximations we are making, it is certainly not our final goal. Our interest is in inserting these expressions for g'_h into (5.27) to get u'^h, and subsequently inserting that approximation of the fine-scale solution back into our coarse-scale equation (5.35) to obtain $\bar{u} = u^h$.

Constant velocity example

Let us consider the simplest case where (5.30) and (5.31) reduce to simply

$$au_{,x} - \kappa u_{,xx} = 0 \quad \text{on } (0,1), \tag{5.56}$$

$$u(0) = 0, \tag{5.57}$$

$$u(1) = 1, \tag{5.58}$$

$$\tag{5.59}$$

which has the exact solution

$$u(x) = \frac{\left(e^{\frac{a}{\kappa}x} - 1\right)}{\left(e^{\frac{a}{\kappa}} - 1\right)}. \tag{5.60}$$

We will seek a numerical solution on a mesh comprised of five linear elements for the advection-dominated case of $a = 1$ and $\kappa = 10^{-6}$. We begin by modeling g'_h with a single linear element. The result is shown in Figure 5.16. Already the solution is nodally exact to within five decimal places of the exact solution. This level of accuracy is maintained through refinements of the fine-scale space, as we would expect.

Let us now consider the same equation, but with a coarse-scale basis of five quadratic elements, and with $a = 1$ and $\kappa = 10^{-2}$. Again we will seek H^1-optimality. This results in the end-nodes of each element interpolating the exact solution, but now we may lose monotonicity as the mid-nodes adjust to decrease the error further. Figure 5.17 shows the results for the fine scales approximated using quadratics on each element with a sub-mesh of 1, 5, and 10 elements. The first 1 sub-element and 5 sub-element curves show some error at the end-nodes, though it is certainly not large. No such error is visible in the 10 sub-element case. To further investigate the quality of the results using the 10 element sub-mesh, Figure 5.18 compares our solution with an H^1-fit of the exact solution using the same coarse-scale five-quadratic-element basis. The two curves are virtually indistinguishable.

Note that monotonicity is frequently a property one seeks in a method when the exact solution is known to be monotone. We have just seen a case where H^1-optimality does not result in a monotone approximation even though the curve being fit is monotone. Thus it is reasonable to ask, "is H^1-optimality a good condition to be seeking?" The answer to this question is not clear,

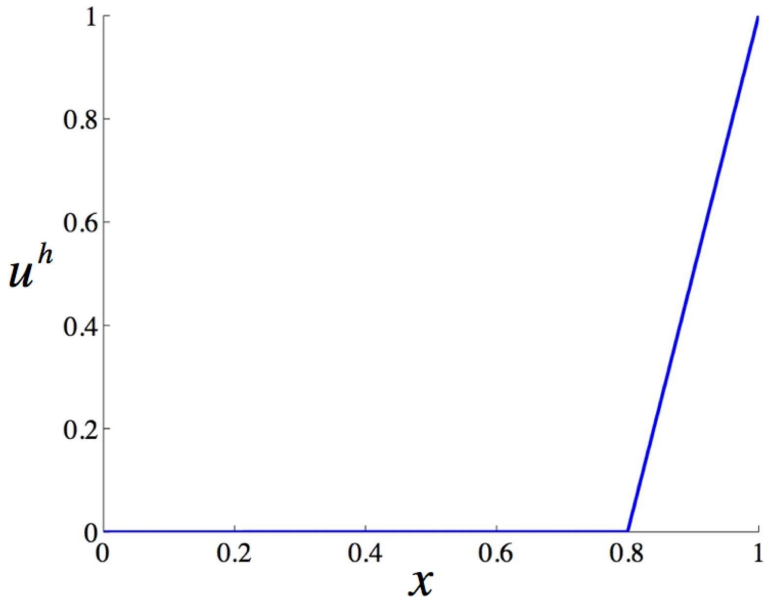


Figure 5.16: 1D advection-diffusion. The coarse-scale solution (shown) is given by 5 linear elements. The fine scales were modeled on each element by the same linear basis.

but work does exist on seeking solutions to certain classes of problems that are optimal in norms that are not induced by inner products (see, *e.g.*, Geurmond [27]). One reason that we focus on H^1-optimality is that the H^1-norm is induced by an inner product and does not require the solution to a nonlinear problem when the equation being solved is linear. Another reason is that it was seen in [39] that the fine-scale Green's function corresponding to the H^1-projection is more highly attenuated (even in $d > 1$) than is the fine-scale Green's function corresponding to the L^2-projection. This implies a greater chance of success when making local approximations to globally supported functions such as g'.

Non-constant velocity example

Let us briefly return to the problem of Section 4.2.2 given by (4.37)-(4.39) as this is a case in which SUPG is not nodally exact (or even monotone). For $\kappa = 10^{-6}$, Figure 5.19 shows our solution using a mesh of 20 linear elements to represent the coarse scales and one linear element to represent the fine scales on each coarse element. The maximum error is less than 0.02% at the nodes. Recall that such accuracy in Section 4.2.2 required the use of a function g' that needed 300 digits of precision to be evaluated properly. Figure 5.20 shows the result using 20 quadratic elements for the coarse scales and 1 quadratic element for the fine scales on each coarse-scale element. Again, the result is for all intents and purposes exact at the end-nodes of each element, but it is not monotone.

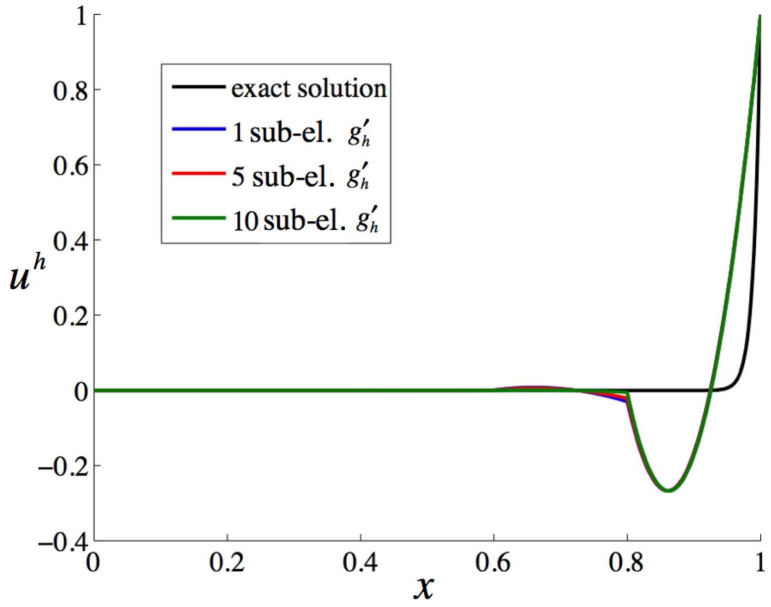
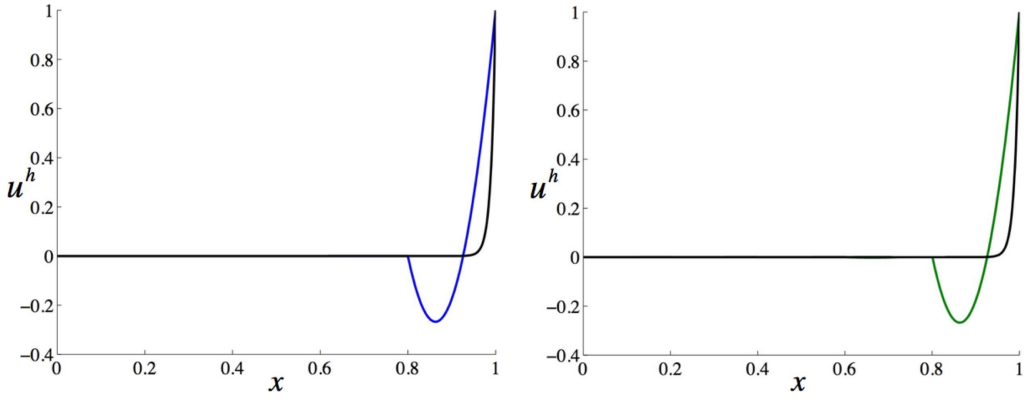


Figure 5.17: 1D advection-diffusion. The coarse-scale solutions are given by 5 quadratic elements. The fine scales were modeled on each element by the number of quadratic sub-elements indicated in the figure.



(a) (b)

Figure 5.18: H^1-optimality with quadratic elements. a) Our numerical solution on a 5 element mesh. b) An H^1-optimal curve-fitting of the exact solution using the same 5 element basis.

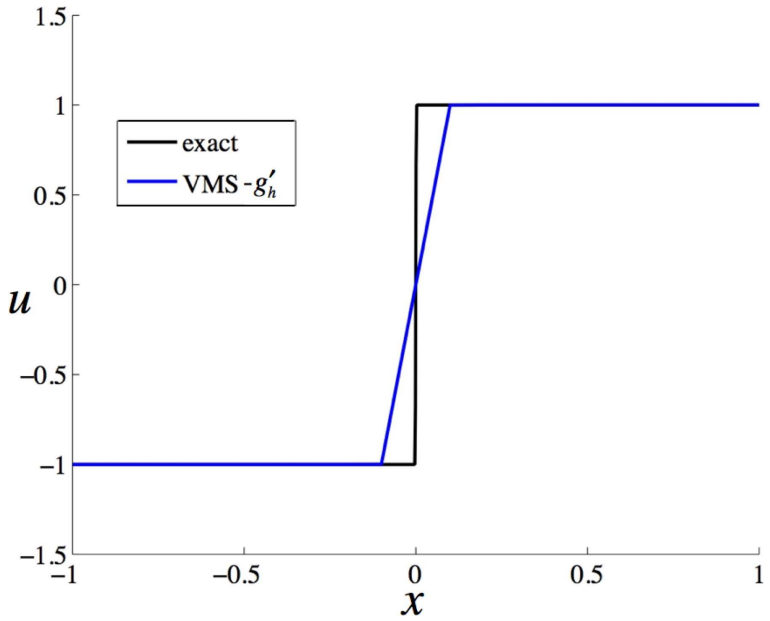


Figure 5.19: Example with non-constant velocity. The coarse-scale solutions are given by 20 linear elements. The fine scales were modeled on each element by the same linear basis.

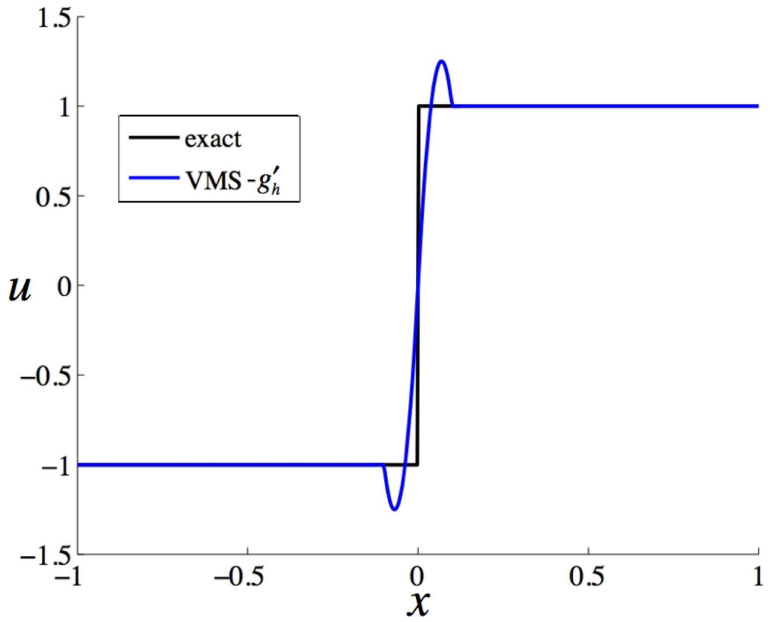
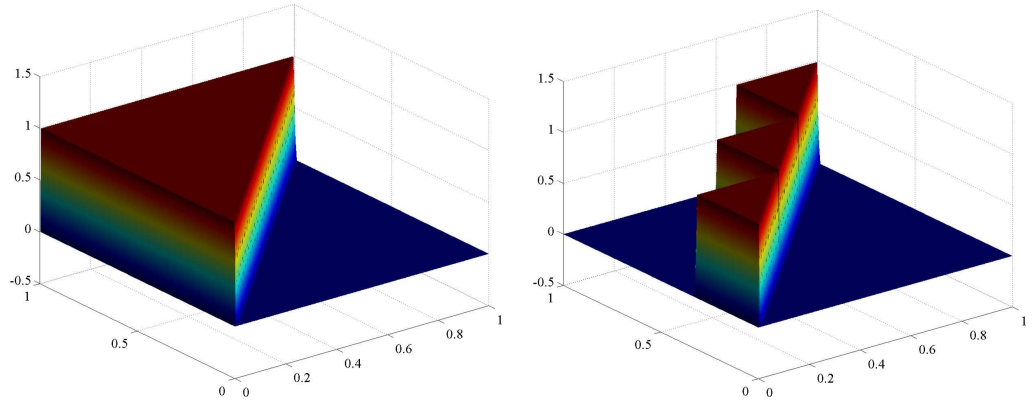


Figure 5.20: Example with non-constant velocity. The coarse-scale solutions are given by 20 quadratic elements. The fine scales were modeled on each element by the same quadratic basis.

The numerical projector in 1D

Throughout this section we have avoided constructing a numerical projector as in (5.21). In the case of linear coarse scale elements, we implicitly used the H^1-projector by taking $g'_h = g^e_h$. For the quadratic case, we used the analytical developments of [39] to get a formula for g'_h from g^e_h. Throughout, however, we could have followed the general approach of Section 5.2. The biggest advantage of *not* explicitly building and using a numerical projector is that it requires that the fine-scale space be richer than the coarse-scale space. If the two spaces are identical, then $\mathbf{P}$ becomes simply the identity and thus $\mathbf{G}' = \mathbf{G} - \mathbf{G}\left(\mathbf{G}\right)^{-1}\mathbf{G} = \mathbf{0}$. Still, while we are in the setting where fine-scale Green's functions are still relatively easy to compute analytically, it is interesting to explore all of the machinery that we have developed.

Let us consider a global coarse-scale mesh of 3 linear elements, $\alpha = 10^6$ (constant velocity). Figure 5.21a shows the analytically computed global Green's function, while Figure 5.21b shows the analytically computed fine-scale Green's function. Using a sub-mesh of 5 linear fine-scale elements per coarse-scale element, we get the global and fine-scale numerical Green's functions shown in Figures 5.22a and 5.22b, respectively. As our weak boundary conditions were applied at the patch boundary (in this case the entire domain), the boundary condition on the global problem is practically ignored at this grid Peclet number. Figure 5.23 shows g'_h restricted to a single element.

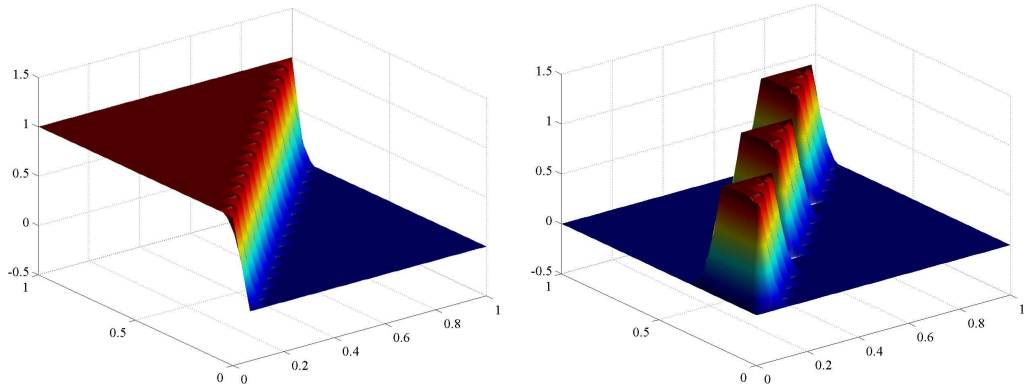


(a) (b)

Figure 5.21: Analytically computed Green's functions for a coarse mesh of 3 linear elements. a) Global Green's function $g(x, y)$. b) Fine-scale Green's function $g'(x, y)$.

5.3.2 2D examples

In this section we will consider the advection-skew-to-mesh problem discussed previously in Section 3.2. The problem setup is described in Figure 3.17. For all of the forthcoming examples we



(a) (b)

Figure 5.22: Numerically computed Green's functions for a coarse mesh of 3 linear elements. The fine-scale mesh has 5 sub-elements for every coarse-scale element. a) Global Green's function $g_h(x, y)$ b) Fine-scale Green's function $g'_h(x, y)$

will take $\theta = \tan^{-1}(2)$, and we will consider $|\mathbf{a}| = 1$ and $\kappa = 10^{-6}$ except where otherwise stated. We use $s = 1$ and $\epsilon = 6$ in all cases. As we will always use a sub-mesh in this section, all of the examples will be solved using the VMS-τ formulation of the fine-scale problem to ensure stability.

When we consider working in spaces with dimension higher than one, there is no orthogonal projector that allows us to localize our fine-scale problem to the element as we did in 1D. Here we must pose (5.36) on a patch as in Figure 5.1. As stated above, we would ideally pose the problem globally, but this could get quite expensive if a fine sub-mesh is used.

Heuristic evidence indicates that we may be able to capture the major features of the fine-scale Green's function with an approximation that is local to the patch. Figure 5.24a shows a globally computed approximation to the global Green's function for a coarse mesh of 25 linear elements, with a fine mesh that partitions each coarse element into 25 sub-elements. Here we consider $\mathbf{y}^* = (0.75, 0.75)$ and plot $g_h(\mathbf{x}, \mathbf{y}^*)$. Figure 5.24b shows a locally computed approximation to that same global Green's function calculated by posing (5.36) on a 3×3 patch. There is quite a bit of difference between the two, and much of the "tail" of the globally computed function is missing from our local approximation (which implicitly assumes g_h to be zero outside of the patch). Fortunately, it is not g_h that we are interested in, but g'_h. Figure 5.25a shows a globally computed approximation to the fine-scale Green's function calculated using $\mathcal{P} = \mathcal{P}_{H^1}$, the H^1-projector. Note that the fine-scale Green's function is *much* more highly attenuated than the global Green's function. Furthermore, the locally computed fine-scale Green's function, shown in Figure 5.25b, differs very little within the patch from that in Figure 5.25a. Again, the fine-scale Green's function seems to be highly attenuated *outside* of Ω_e^P, and the approximations *inside* the patch are very similar regardless

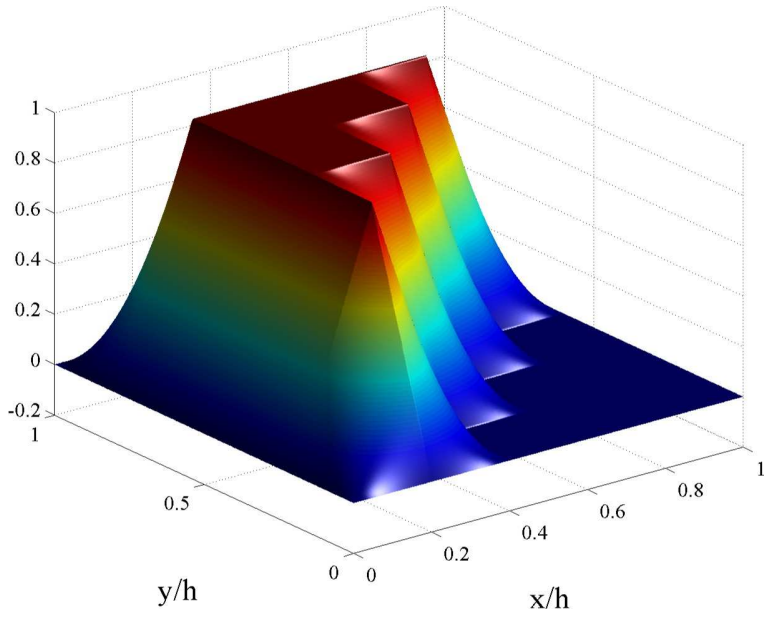


Figure 5.23: Numerically computed Green's functions for a coarse mesh of 3 linear elements. The fine-scale mesh has 5 sub-elements for every coarse-scale element. Detail of g'_h restricted to a single element.

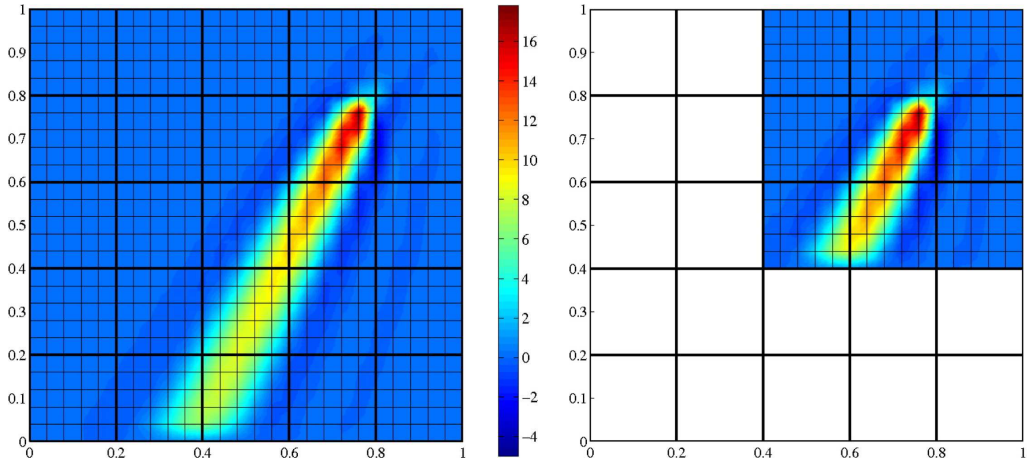
of whether they came from a global fine-scale problem or a local one.

The choice of the projector has a profound impact on the quality of the results. Though there is no analytical expression for the solution to our two-dimensional problem, we have constructed *pseudo-optimal* results to compare with by using a surface-fitting algorithm to find L^2- and H^1-optimal coarse-mesh approximations to a fine-mesh solution given by SUPG on a 200×200 mesh[3]. The results are shown in Figures 5.26 and 5.27 for a 20×20 mesh of linear elements. Note that the H^1-optimal solution contains fewer overshoots and undershoots, but that neither solution is monotone.

We obtain computational results by following exactly the procedure discussed in Section 5.2. First we choose a patch size (given in terms of the number of coarse-scale elements in each direction – always an odd number so that each element Ω_e is at the center of its patch Ω_e^P) and a fine-scale mesh resolution (given in terms of the number of sub-elements in each direction that a single coarse element is divided into). For example, a 3×3 patch with a 5×5 sub-mesh (as seen in Figure 5.25) has 225 fine-scale elements per patch: 25 in each of its 9 coarse-scale elements.

To obtain a system of algebraic equations which may be solved for $\bar{u} \equiv u^h$, we loop through

[3]Actually, the SUPG solution still had minor overshoots and undershoots, which are known to be non-physical. We post-processed the fine-mesh solution by rounding any value greater than one back down to one, and rounding any value less than zero back up to zero.



(a) (b)

Figure 5.24: Approximations to the global Green's function, $g_h(\mathbf{x}, \mathbf{y}^*)$, where $\mathbf{y}^* = (0.75, 0.75)$, for a 5×5 linear coarse-scale mesh with each element partitioned into a 5×5 sub-mesh. a) Global approximation over the entire domain. b) Local approximation on a 3×3 patch.

the coarse-scale elements, assembling them one at a time. On each element Ω_e we begin by building $\mathbf{K}'$ over the patch as in (5.45), which we invert to get $\mathbf{G}$ as in (5.14). Next, we build the projection matrix $\mathbf{P}$ as in (5.21) where our choice of inner product defines which projection we are using. Inserting $\mathbf{G}$ and $\mathbf{P}$ into (5.26), we obtain $\mathbf{G}'$, which gives us g'_h as in (5.25). We then insert g'_h into (5.27) to arrive at an expression for u'^h that may be inserted into our coarse-scale equation (5.35). Finally, we assemble (5.35) into the global system and move on to the next element.

Note that we never really build u'^h explicitly. Due to the separable form of $g'_h(\mathbf{x}, \mathbf{y})$, we can treat the term $\left(\mathcal{L}^*\bar{w}, u'^h\right)_{\Omega_e}$ as

$$\begin{aligned}
\left(\mathcal{L}^*\bar{w}, u'^h\right)_{\Omega_e} &= \int_{\Omega_e} \mathcal{L}^*\bar{w}(\mathbf{y}) u'^h(\mathbf{y}) d\mathbf{y} \\
&= \int_{\Omega_e} \mathcal{L}^*\bar{w}(\mathbf{y}) \left(\int_{\Omega_e^P} N'_\alpha(\mathbf{y}) \left[\mathbf{G}'\right]_{\alpha\beta} N'_\beta(\mathbf{x}) R'(\bar{u}(\mathbf{x})) d\mathbf{x} \right) d\mathbf{y} \\
&= \left(\int_{\Omega_e} \mathcal{L}^*\bar{w}(\mathbf{y}) N'_\alpha(\mathbf{y}) d\mathbf{y} \right) \left[\mathbf{G}'\right]_{\alpha\beta} \left(\int_{\Omega_e^P} N'_\beta(\mathbf{x}) R'(\bar{u}(\mathbf{x})) d\mathbf{x} \right) \\
&= \left(\mathcal{L}^*\bar{w}, N'_\alpha\right)_{\Omega_e} \left[\mathbf{G}'\right]_{\alpha\beta} \left(N'_\beta, R'(\bar{u})\right)_{\Omega_e^P}. \qquad (5.61)
\end{aligned}$$

This equation is linear with respect to both $\bar{w}$ and $\bar{u}$, as we would hope. The only additional com-

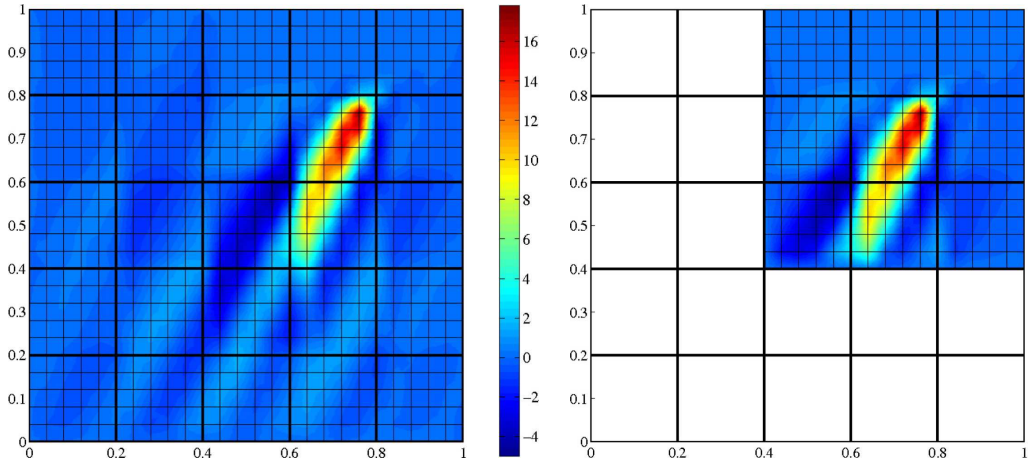


Figure 5.25: Approximations to the fine-scale Green's function, $g'_h(\mathbf{x}, \mathbf{y}^*)$, where $\mathbf{y}^* = (0.75, 0.75)$, for a 5×5 linear coarse-scale mesh with each element partitioned into a 5×5 sub-mesh. a) Global approximation over the entire domain. b) Local approximation on a 3×3 patch.

plexity in terms of solving the global system will come from the fact that (5.61) couples each function with support on element Ω_e to every other function with support on patch Ω_e^P. Thus the global system will be more dense than the corresponding Galerkin's method problem on the same coarse mesh.

Let our coarse scales be given by the same 20×20 linear mesh as used for our pseudo-optimal fitting above. We consider several combinations of patch size and mesh resolution, calculating the error with respect to our 200×200 reference solution in the L^2-norm and the H^1-seminorm for the cases of L^2- and H^1-projection, respectively. Results are shown in Tables 5.1 and 5.2. Not surprisingly, results improve when either the patch-size is increased or the number of sub-elements is increased. Experience shows that the case of a 3×3 patch with a 5×5 sub-mesh is the most efficient option as it is more accurate than the 5×5 patch with a 3×3 sub-mesh and markedly faster, albeit slightly less accurate, than the 5×5 patch with a 5×5 sub-mesh. Visually, the results for all cases are fairly hard to distinguish. Figure 5.28 shows our best result (the $5 \times 5, 5 \times 5$ case) for the L^2-projection. This compares very favorably with our L^2 pseudo-optimal result in Figure 5.26. Similarly, Figure 5.29 shows our best result for the H^1-projection. Comparison with the H^1 pseudo-optimal result of Figure 5.27 is also favorable, and the difference between the two projections is clear.

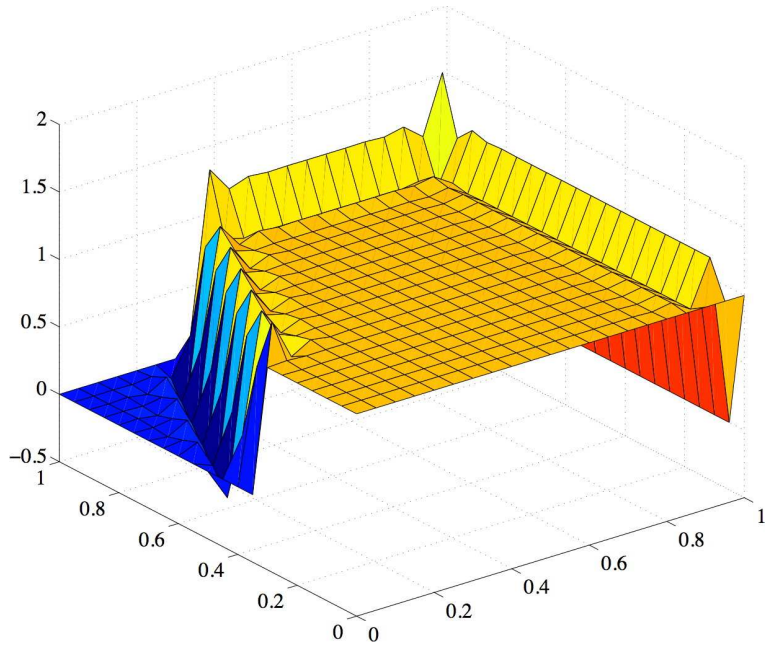


Figure 5.26: L^2-fitting. Pseudo-optimal solution generated by projecting a 200×200 linear element fine-mesh solution onto a 20×20 linear element mesh.

Connections to previous work

Certain aspects of the present method are evocative of techniques already existing in the literature. The connection between nodal exactness in 1D and the existence of a Green's function in the space spanned by the weighting functions dates back to at least 1973 when Strang and Fix attributed the idea to Douglas and Dupont (see [52], p. 168). Our use of a fine-scale mesh to compute local approximations to fine-scale Green's functions bears a resemblance to the locally supported "numerically optimal test functions" constructed on "microelements" by Demkowicz and Oden [21] in 1986. The fundamental point of departure between our approach and such previous techniques is our insistence on adherence to the variational multiscale method, whereas these other approaches are rooted in (and crippled by) a blind adherence Galerkin's method.

The issue of practicality

The developments of this chapter have served several immediate purposes. First, a connection has been made between the multiscale discontinuous Galerkin method and the variational multiscale method that puts the former into its appropriate context. Second, a technique has been devised for approximating the fine-scale Green's function in VMS. This is an entity that has existed as a concept

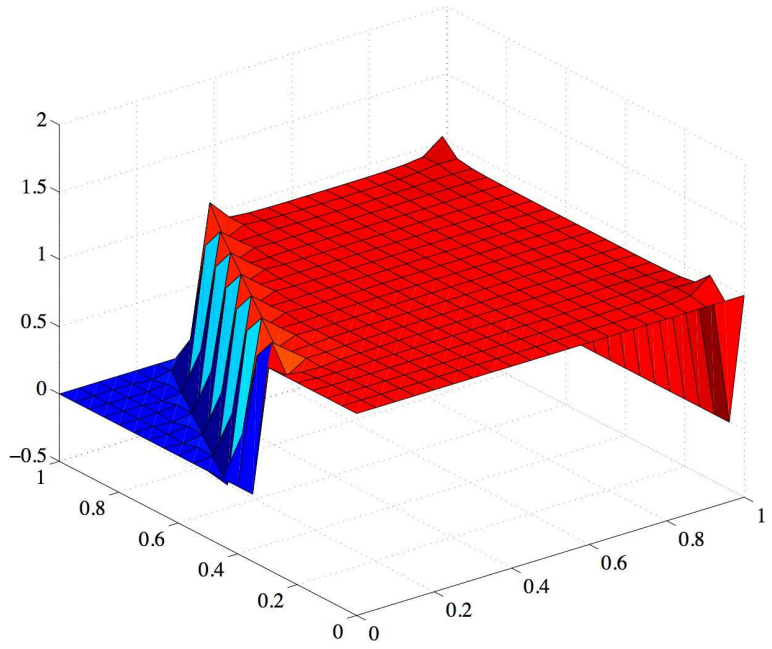


Figure 5.27: H^1-fitting. Pseudo-optimal solution generated by projecting a 200×200 linear element fine-mesh solution onto a 20×20 linear element mesh.

in the literature since the mid 1990's, only having been examined thus far in the limited number of cases that are analytically tractable in the 2005 work of Hughes and Sangalli [39]. For the first time, we are able to examine these functions through numerical means in a general setting, and this provides a unique opportunity to learn about the nature of the fine scales as they arise in VMS. Lastly, we are able to use these techniques to build numerical methods that seek to approximate the solutions of differential equations in the norm of our choosing[4].

While this last achievement is valid, it may not be the most important at this moment. The computational costs of this method are quite large in comparison with classical stabilized methods such as SUPG and VMS-τ. However, our ability to examine these fine scales in a detail that was heretofore impossible opens up new avenues of research into their modeling, and the modeling of their effect on the coarse scales. There is much to be learned from this method, even if it is not appropriate for large-scale computations in its current implementation.

[4] so long as that norm is induced by an inner product.

Patch size / Sub-mesh size	3×3	5×5	7×7
		$(*10^{-2})$	
3×3	2.31275	2.29018	2.28656
5×5	2.23833	2.22575	
7×7	2.22577		

Table 5.1: Error in the L^2 norm relative to the L^2 pseudo-optimal result. The L^2-projection was used in building g'_h. Empty values in the table reflect cases deemed impractically expensive to compute.

Patch size / Sub-mesh size	3×3	5×5	7×7
		$(*10^{-3})$	
3×3	2.79549	2.78400	2.78364
5×5	2.77866	2.76815	
7×7	2.77352		

Table 5.2: Error in the H^1 semi-norm relative to the H^1 pseudo-optimal result. The H^1-projection was used in building g'_h. Empty values in the table reflect cases deemed impractically expensive to compute.

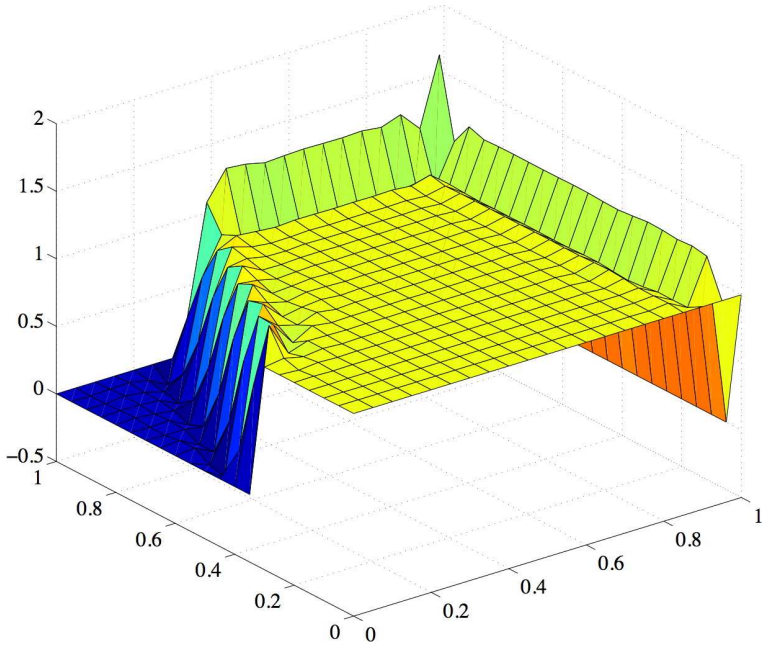


Figure 5.28: Numerical VMS results using the L^2-projection. Compare with Figure 5.26.

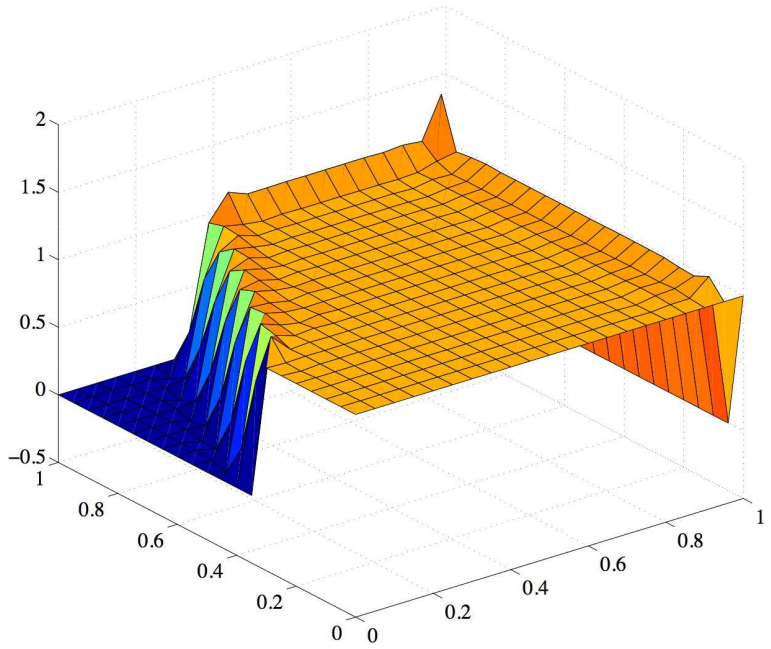


Figure 5.29: Numerical VMS results using the H^1-projection. Compare with Figure 5.27.

Chapter 6

The Parameter-Free Variational Multiscale Method

One of the most important features of the variational multiscale method in its purest form is that it is *exact*. The equation for the coarse-scale solution is exact given an expression for the fine-scale solution. The equation for the fine scales is exact, but of infinite dimension. Though typically intractable in such a form, having this strong theoretical foundation allows us to carefully identify exactly what approximations we are making as we seek a numerical solution. It also allows us to interpret (as we did previously for MDG and Galerkin's method) the approximations that we make, which itself provides us a direction in which to seek future improvements. This is what distinguishes it from previous stabilized methods: we begin with an idealization and work backwards making carefully articulated approximations until we have a method which is approximate but tractable. Classical stabilized methods begin with Galerkin's method – a method that is tractable but that might metaphorically be considered to be "sick" – and work forward by addressing the "symptoms" (spurious oscillations) instead of the "disease" (the fact that the unresolved scales are completely ignored).

The numerical approach to VMS described in Chapter 5 seeks to take the former approach. We find numerical solutions within a VMS framework while preserving as much of the exact analytical structure as possible, and we have an idea of the path to follow to improve the method (*e.g.*, larger patches and richer fine-scale spaces). In many ways, we consider this approach a great success. The price that has been paid, however, is in computational expense. The method is still very slow to run, even for fairly modest patch sizes.

In this chapter, we strip away more of the VMS machinery, and we introduce a simplified version of the method that is both fast and robust. While it lacks some of the theoretical foundation of the techniques from the previous chapter, it generalizes in a natural way that makes it more far-

reaching than previous stabilized methods. In particular, the length scales and problem parameters that must be arbitrarily combined in the design of stabilization parameters for SUPG and VMS-τ are naturally incorporated into this approach, regardless of the spatial dimension or shape of the elements. This makes extension of the method to new problems far simpler than for previous stabilized methods.

6.1 A simple multiscale scheme: g_h^e revisited

In Sections 5.1.1 and 5.1.2, we describe how the multiscale discontinuous Galerkin method is implicitly using an element Green's function to obtain an approximation to the fine scales. In MDG, this concept was never articulated. Instead, an "interscale transfer operator" was constructed that allowed for the solution of a global DG system with the number of degrees-of-freedom as a CG method. The expression returned as the solution to the problem, however, was discontinuous, and the coarse-scale equation was that of a DG formulation. Here, we propose using the element Green's function to obtain a local approximation to the fine-scale field, but using the resulting expression for u'^h in the coarse-scale equation emanating from a CG formulation. The result of the numerical method will be the coarse-scale field, with u'^h used only to approximate the effect of the fine scales on the coarse scales. We call this approach the ***parameter-free variational multiscale*** (PVMS) method.

This method is just a vastly simplified version of the method from the previous chapter. We use exactly the same fine-scale problem as before, (5.13), taking our patch to be a single element ($\Omega_e^P \equiv \Omega_e$). Locally, we use a polynomial basis to represent the fine scales (*i.e.*, no sub-mesh), thus the amount of additional quadrature over that necessary for coarse-scale assembly is minimal. Even if we consider a higher polynomial order for the fine scales, the matrix inversion, (5.14), involves a matrix much smaller than that required for a problem posed over even a small patch. No projections are needed, saving the expense of both building and applying them. As we are not using a sub-mesh, there is no need to stabilize the fine-scale problem, thus eliminating the need to select a stabilization parameter. Lastly, as our coarse-scale basis functions are no longer coupled through the fine-scale Green's function, the sparsity of our global matrix is identical to that for Galerkin's method.

6.2 A linear example: the advection-diffusion equation

In Section 5.3.1, Figures 5.16 and 5.19 depict results using this method. Recall that in 1D H^1-optimality for a linear basis dictated that $g_h' \equiv g_h^e$. Furthermore, such good results were obtained for a single element approximation of g_h^e that no richer basis was considered (when the coarse scales were linear). Thus, we have already seen PVMS in action in the special case where our general

approach reduced to this simplified one.

In two dimensions, let us again consider the problem of Section 5.3.2. Figures 6.1 and 6.2 show solutions using linear and quadratic fine-scales, respectively. Though both results are at least as good as SUPG, the quality improvement in going to the quadratic basis is evident, and the difference in speed for such a coarse mesh is negligible. Still, there are reasons why using linears for the fine scales might be preferable. First, use of the same basis allows for the elimination of certain redundant calculations (the difference between the element stiffness matrix for the fine scales and the coarse scales is in the boundary integral, the other terms are shared between them). Second, the use of quadratics necessitates using a higher quadrature rule, which may prove excessively expensive in nonlinear, time-dependent applications.

If one desires an improved solution using the linear fine-scale basis, there is one option worth considering. The use of weakly imposed boundary conditions on the fine-scale problem has allowed us to be successful while using such crude approximations. It makes sense for us also to consider using weakly imposed boundary conditions on the global coarse-scale problem as well. Using the formulation of Bazilevs and Hughes [7], representing the coarse and fine scales with the same linear basis (at the element level), and post-processing the result by overwriting the boundary degrees-of-freedom with the exact boundary condition, we arrive at the result shown in Figure 6.3. We consider this result to be very close to perfect. The expense of our method is comparable to that of any other stabilized method, we have avoided any of the standard parameter selection dilemmas, and we have obtained a result that is as accurate, while being free from spurious oscillations, as any linear method that we are aware of.

6.3 Nonlinear examples

We can extend this approach to nonlinear applications by applying the same technique to the linearized problem appearing within each iteration of a Newton-Raphson solver. In this way, we can address a much more general class of problems.

Consider an abstract nonlinear problem for which we seek a variational multiscale formulation. The equation may contain several distinct nonlinear terms (or a combination of linear and nonlinear terms). For each such term, our approach is to expand it in a Taylor series about the coarse scales. For instance, let $\mathbf{N}(u)$ be a nonlinear operator acting on u. Then for $u = \bar{u} + u'$ we have

$$\begin{aligned}\mathbf{N}(\bar{u}+u') &= \mathbf{N}(\bar{u}+\epsilon u')\big|_{\epsilon=0} + \frac{d}{d\epsilon}\mathbf{N}(\bar{u}+\epsilon u')\bigg|_{\epsilon=0} + \frac{1}{2}\frac{d^2}{d\epsilon^2}\mathbf{N}(\bar{u}+\epsilon u')\bigg|_{\epsilon=0} + \dots \\ &= \mathbf{N}(\bar{u}) + \frac{d\mathbf{N}(\bar{u})}{du}u' + \frac{1}{2}\frac{d^2\mathbf{N}(\bar{u})}{du^2}(u')^2 + \dots . \end{aligned} \tag{6.1}$$

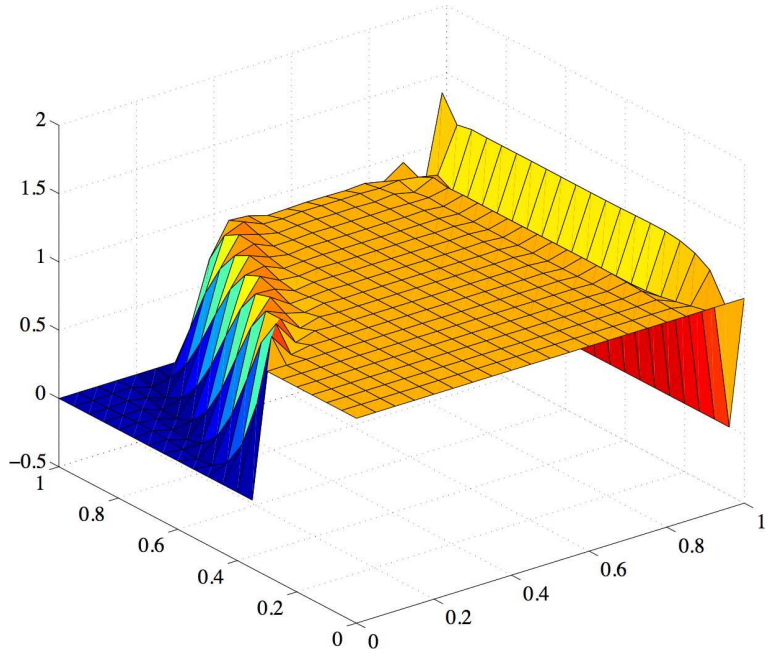


Figure 6.1: Numerical VMS results using linears to represent g_h^e on each element.

In practice, we will only keep as many terms as we need for our desired level of accuracy and stability. Furthermore, the number of terms in the series that we keep does not need to be the same for every term in the original equation. For instance, we may choose to keep more of the expansion of the advective term than of the time derivative term as the former seems more culpable in the generation of instabilities.

6.3.1 Burgers' equation

Let us consider the unforced 1D Burger's equation on $\Omega = (-1, 1)$ with homogeneous boundary conditions. The strong form of the equation seeks u such that:

$$u_{,t} - \left(\frac{u^2}{2}\right)_{,x} - \lambda u_{,xx} = 0 \quad \text{in } \Omega, \tag{6.2}$$

$$u(-1, t) = u(1, t) = 0, \tag{6.3}$$

$$u(x, 0) = u_0(x). \tag{6.4}$$

Multiplying by weighting function w and integrating yields the following weak form: find $u \in V$ such that

$$a(w, u) = 0 \quad \forall w \in V, \tag{6.5}$$

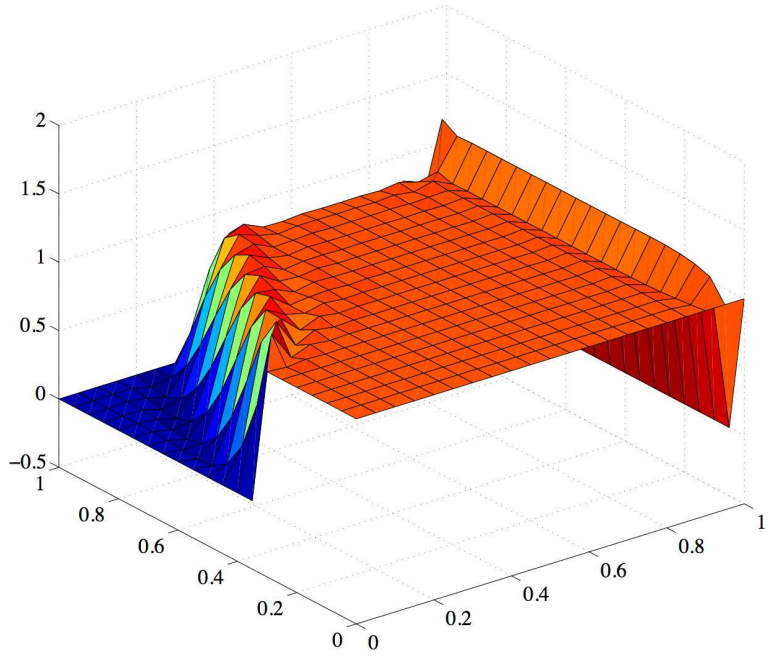


Figure 6.2: Numerical VMS results using quadratics to represent g_h^e on each element.

where our semilinear form is given by

$$a(w,u) = (w,u_{,t})_\Omega - \left(w_{,x}, \frac{u^2}{2}\right)_\Omega + (w_{,x}, \lambda u_{,x})_\Omega \,. \tag{6.6}$$

We now split the solution space into coarse and fine components, as in (4.3). Thus, with $u = \bar{u} + u'$, we get the following coarse-scale problem[1]: given $u' \in V'$, find $\bar{u} \in \bar{V}$ such that

$$a(\bar{w},\bar{u}) + a(\bar{w},u') - \left(w_{,x}, \bar{u}u'\right)_\Omega = 0 \quad \forall \bar{w} \in \bar{V}. \tag{6.7}$$

We write our fine-scale problem on the element as

$$\begin{aligned} &\left(w', u'_{,t}\right)_{\Omega_e} - \left(w'_{,x}, \bar{u}u' + \frac{(u')^2}{2}\right)_{\Omega_e} + \left(w'_{,x}, \lambda u'_{,x}\right)_{\Omega_e} \\ &+ \left(w'\left(\bar{u}u' + \frac{(u')^2}{2}\right)\chi_{out} - \lambda w' u'_{,x} + s\lambda w'_{,x}u'\right)\Big|_{x_e}^{x_{e+1}} \\ &\qquad + \left(\frac{\epsilon\lambda}{h}u'\right)\Big|^{x_e,x_{e+1}} = \left(w', R'(\bar{u})\right)_{\Omega^e}, \end{aligned} \tag{6.8}$$

[1]We have implicitly used our proposed Taylor expansion. The nonlinear term $N(\bar{z}+z') = (\bar{z}+a')^2/2$ has an expansion of just three terms, each of which we have kept thus far.

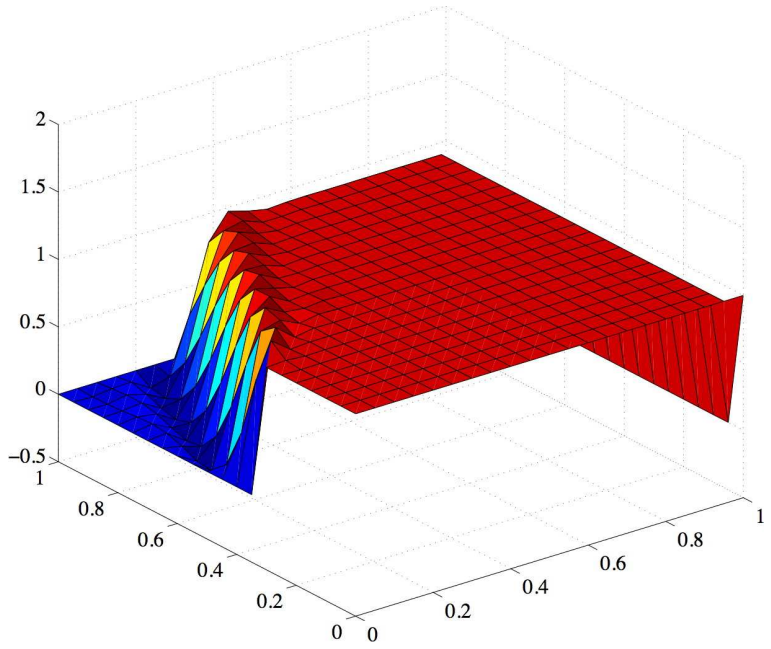


Figure 6.3: Numerical VMS results using linear fine-scale representations with weakly enforced boundary conditions. The result for u^h on the boundary is overwritten with the exact Dirichlet data.

where $f(z)|_a^b = f(b) - f(a)$ and $f(z)|^{a,b} = f(a) + f(b)$. Note that the solution itself plays the role of the velocity here, so with normal $n = -1$ on x_e and $n = 1$ on x_{e+1} we have

$$\chi_{out} = \begin{cases} 1 & \bar{u}n > 0, \\ 0 & \text{otherwise}. \end{cases} \tag{6.9}$$

Two major differences are immediately evident when comparing (6.8) with the fine-scale equations corresponding to linear problems that we have seen (*e.g.*, (5.36)). The first is that the coarse-scale solution appears on the left side of this equation, not simply on the right. We are going to be solving (6.7) by using a backward Euler solver in time and a Newton-Raphson iterative scheme within each time step. Thus, at every time step n and iteration k, we have an expression for $\bar{u}$ denoted $\bar{u}_n^k$, and anywhere that $\bar{u}$ appears in (6.8) we treat it as data and use our most recent known solution $\bar{u}_n^k$. The other major difference from our previous cases is that (6.8) is nonlinear with respect to u'. Though several approaches to dealing with this term are possible (*e.g.*, using an iterative method to solve the fine-scale equation, storing the fine-scale solution and lagging one of the terms such that $(u_n'^k)^2 \approx u_n'^{k-1} u_n'^k$, etc.), we will take the simplest possible approach. We assume that our fine scales are small and as such $u_n'^k \ll 1 \Rightarrow (u_n'^k)^2 \lll 1$, thus we simply omit the

terms that are quadratic in the fine scales. This leaves us with the following problem to solve for $u_n'^k$, the fine-scale solution corresponding to $\bar{u}_n^k$: find $u_n'^k \in V'(\Omega_e)$ such that

$$\begin{aligned}\left(w', u_{n,t}'^k\right)_{\Omega_e} - \left(w'_{,x}, \bar{u}_n^k u_n'^k\right)_{\Omega_e} + \left(w'_{,x}, \lambda u_{n,x}'^k\right)_{\Omega_e} \\ + \left(w'\left(\bar{u}_n^k u_n'^k\right)\chi_{out} - \lambda w' u_{n,x}'^k + s\lambda w'_{,x} u_n'^k\right)\Big|_{x_e}^{x_{e+1}} \\ + \left(\frac{\epsilon\lambda}{h} u_n'^k\right)\Big|^{x_e, x_{e+1}} = \left(w', R'(\bar{u}_n^k)\right)_{\Omega^e}. \end{aligned} \tag{6.10}$$

As this problem is linear and posed on the element, it is not expensive to solve.

Assuming that we are using linear coarse-scale elements, we can further simplify parts of our coarse-scale equation. If we integrate by parts (with $u'|_\Omega = 0$ due to the Dirichlet conditions) we have

$$a(\bar{w}, u') = \left(\bar{w}, u'_{,t}\right)_\Omega - \left(\bar{w}_{,x}, \frac{(u')^2}{2}\right)_\Omega - \left(\bar{w}_{,xx}, \lambda u'\right)_\Omega. \tag{6.11}$$

We may choose to omit the first term (equivalent to keeping only the zeroth order term in the Taylor expansion of the *linear* time derivative term of (6.2)). As we are using linear elements in space, we assume $\bar{w}_{,xx} = 0$. Lastly, as we have assumed already that $u' \ll 1$, we can assume that the quadratic term is vanishingly small. This leaves us with the the coarse scale equation: given $u' \in V'$, find $\bar{u} \in \bar{V}$ such that

$$a(\bar{w}, \bar{u}) - \left(w_{,x}, \bar{u}u'\right)_\Omega = 0 \quad \forall \bar{w} \in \bar{V}. \tag{6.12}$$

Though these decisions of what and what not to include may seem arbitrary, we are aware of exactly what approximations we have made. If the quality of our solution were not sufficient, we could immediately attempt to improve it by including some of the terms that we have omitted here. The results indicate that what we have kept is sufficient.

Let us solve (6.12) on a mesh of 30 linear elements using PVMS for $\lambda = 10^{-2}$, $t \in [0, 1]$, $\Delta t = .001$, and $u_0(x) = -\sin(\pi x)$. For comparison, a solution using Galerkin's method on the same mesh, a VMS-τ solution with the stabilization parameter artfully chosen to be

$$\tau = \left((\bar{u}_{,x})^2 + \left(\frac{2\bar{u}}{h}\right)^2 + \left(3\lambda\left(\frac{2}{h}\right)^2\right)^2\right)^{-\frac{1}{2}}, \tag{6.13}$$

and an "overkill" reference solution using Galerkin's method on a mesh of 300 linear elements are all examined. Results plotted at $t = 0, 0.1, 0.2, \ldots, 1$ are shown for Galerkin, VMS-τ, PVMS, and the reference solution in Figures 6.4, 6.5, 6.6, and 6.7, respectively. Figure 6.8 shows a detail of all four solutions at $t = 0.5$. We see that Galerkin's method is unstable, VMS-τ is *slightly* unstable, and PVMS is *slightly* overly diffusive. Typically, though, it is considered "safer" to fail to the

diffusive side as it is less likely to prevent convergence of the nonlinear solver. Also, we got this result without having to concoct a perfect τ that is completely irrelevant to any other problem that we might want to solve.

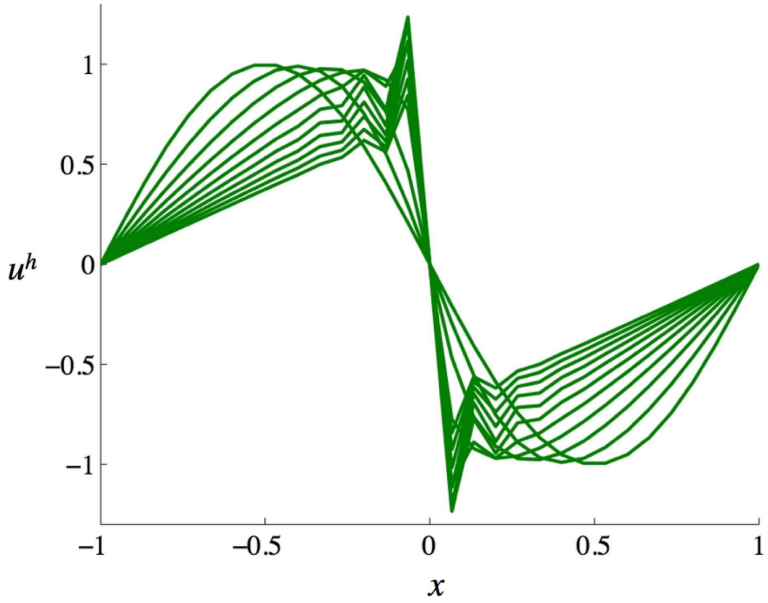


Figure 6.4: Burgers' equation. Galerkin's method on a mesh of 30 linear elements. Solutions shown at $t = 0, 0.1, 0.2, \ldots, 1$.

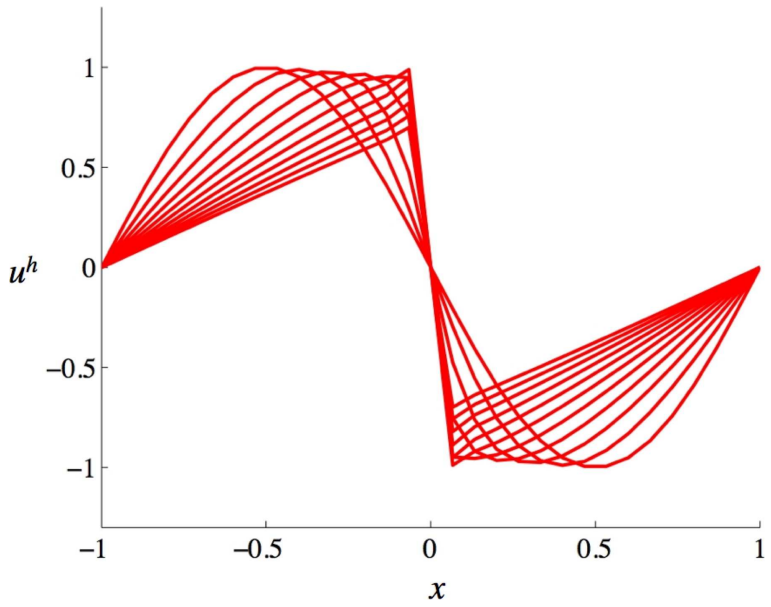


Figure 6.5: Burgers' equation. VMS-τ on a mesh of 30 linear elements. Solutions shown at $t = 0, 0.1, 0.2, \ldots, 1$.

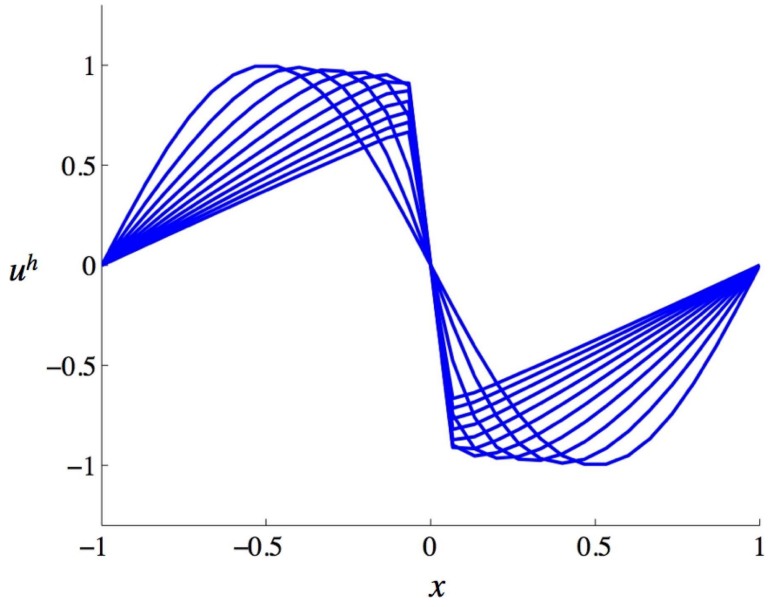


Figure 6.6: Burgers' equation. PVMS on a mesh of 30 linear elements. Solutions shown at $t = 0, 0.1, 0.2, \ldots, 1$.

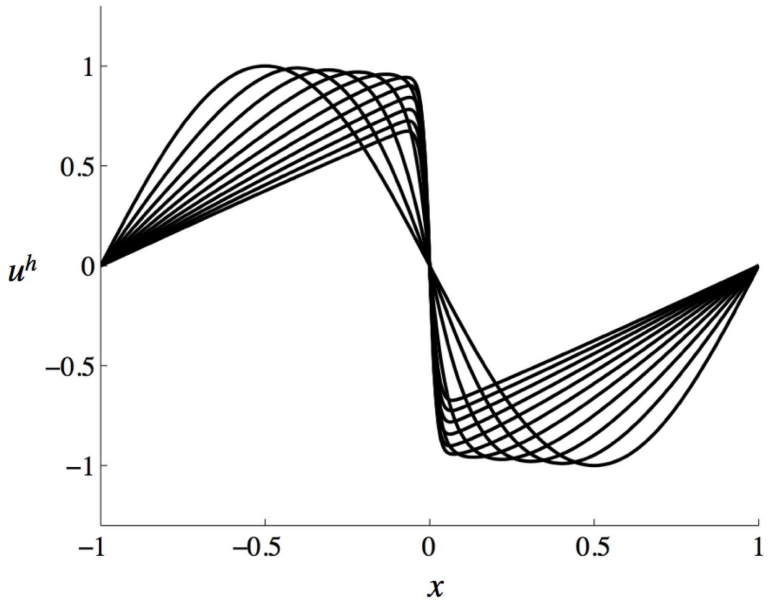


Figure 6.7: Burgers' equation. Reference solution using Galerkin's method on a mesh of 300 linear elements. Solutions shown at $t = 0, 0.1, 0.2, \ldots, 1$.

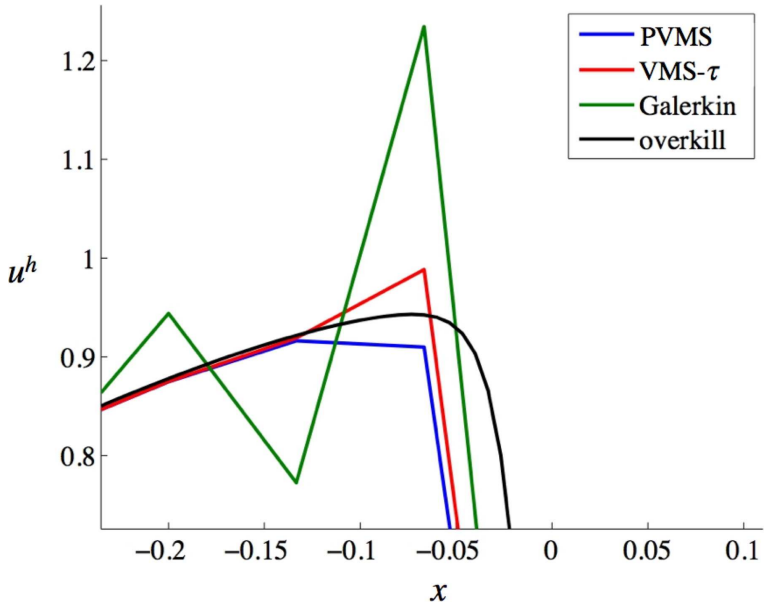


Figure 6.8: Burgers' equation. Detail of all four methods for $t = 0.5$.

6.3.2 The compressible Euler equations

We can expand the method further by considering a nonlinear *system* of equations. Assuming the unforced case, the strong form of the equations in conservation variables is given by

$$\mathbf{U}_{,t} + \mathbf{F}^{\mathrm{adv}}_{i,i} = \mathbf{0}, \tag{6.14}$$

where, in three dimensions,

$$\mathbf{U} = \rho \begin{pmatrix} 1 \\ u_1 \\ u_2 \\ u_3 \\ e \end{pmatrix} \tag{6.15}$$

and

$$\mathbf{F}^{\mathrm{adv}}_i = \rho u_i \begin{pmatrix} 1 \\ u_1 \\ u_2 \\ u_3 \\ e \end{pmatrix} + p \begin{pmatrix} 0 \\ \delta_{1i} \\ \delta_{2i} \\ \delta_{3i} \\ u_i \end{pmatrix} \tag{6.16}$$

are the conservation variables and advective flux, respectively. In the above relations: ρ is the density; u_i is the component of the velocity $\mathbf{u}$ in the direction x_i; e is the total energy density – the sum of the internal energy density ι and the kinetic energy density $\|\mathbf{u}\|^2/2$; p is the thermodynamic pressure, and δ_{ij} is the Kronecker delta. The summation convention is used throughout.

The five equations in (6.14) represent the conservation of mass, momentum, and energy (hence the term "conservation variables"). To solve these equations, we must introduce appropriate *constitutive* relations, such as:

$$\iota = c_v \theta, \tag{6.17}$$

$$p = (\gamma - 1)\rho\iota, \tag{6.18}$$

where c_v is the specific heat at constant volume, θ is the absolute temperature and γ is the ratio of specific heats c_p/c_v (c_p is the specific heat at constant pressure). Equations (6.17) and (6.18) constitute the *ideal gas law*.

We can rewrite these in quasi-linear form using our desired variables, in this case primitive variables $\mathbf{Y} = \{p, u_1, u_2, u_3, \theta\}^{\mathrm{T}}$, as

$$\mathbf{A}_0 \mathbf{Y}_{,t} + \mathbf{A}_i \mathbf{Y}_{,i} = \mathbf{0}, \tag{6.19}$$

where $\mathbf{A}_0 = \mathbf{U}_{,\mathbf{Y}}$ and $\mathbf{A}_i = \mathbf{F}^{\text{adv}}_{i,\mathbf{Y}}$ is the ith Euler Jacobian matrix.

The weak form of (6.19) is given as a semilinear form

$$a\left(\mathbf{W};\mathbf{Y}\right) = \mathbf{0}, \tag{6.20}$$

where

$$a\left(\mathbf{W};\mathbf{Y}\right) \equiv \left(\mathbf{W}, \mathbf{U}_{,t}\left(\mathbf{Y}\right)\right)_\Omega - \left(\mathbf{W}_{,i}, \mathbf{F}^{\text{adv}}_i\left(\mathbf{Y}\right)\right)_\Omega + \left(\mathbf{W}, \mathbf{F}^{\text{adv}}_i\left(\mathbf{Y}\right) n_i\right)_{\partial\Omega}, \tag{6.21}$$

and n_i is the ith component of the normal vector $\mathbf{n}$. After splitting our solution spaces as in (4.3) and inserting $\mathbf{Y} = \bar{\mathbf{Y}} + \mathbf{Y}'$ into (6.20), we expand our nonlinear terms in a Taylor series about the coarse-scale solution, as described at the beginning of this section. In the advective term, we keep everything through the first-order term, truncating the rest:

$$\begin{aligned}
-\left(\mathbf{W}_{,i}, \mathbf{F}^{\text{adv}}_i\left(\bar{\mathbf{Y}} + \mathbf{Y}'\right)\right)_\Omega \approx & -\left(\mathbf{W}_{,i}, \mathbf{F}^{\text{adv}}_i\left(\bar{\mathbf{Y}} + \epsilon\mathbf{Y}'\right)\right)_\Omega\Big|_{\epsilon=0} \\
& - \frac{d}{d\epsilon}\left(\mathbf{W}_{,i}, \mathbf{F}^{\text{adv}}_i\left(\bar{\mathbf{Y}} + \epsilon\mathbf{Y}'\right)\right)_\Omega\Big|_{\epsilon=0} \\
= & -\left(\mathbf{W}_{,i}, \mathbf{F}^{\text{adv}}_i\left(\bar{\mathbf{Y}}\right)\right)_\Omega \\
& - \left(\mathbf{W}_{,i}, \mathbf{A}_i\left(\bar{\mathbf{Y}}\right)\mathbf{Y}'\right)_\Omega,
\end{aligned} \tag{6.22}$$

summing on repeated indices as usual. In our example problem, we will be interested in a steady state solution, so we can be less careful (and therefore require fewer calculations) by keeping only the lowest order in the expansion of the time derivative term (as we did with Burgers' equation, above):

$$\begin{aligned}
\left(\mathbf{W}, \mathbf{U}_{,t}\left(\mathbf{Y}\right)\right)_\Omega &\approx \left(\mathbf{W}, \mathbf{U}_{,t}\left(\bar{\mathbf{Y}}\right)\right)_\Omega \\
&= \left(\mathbf{W}, \mathbf{A}_0\left(\bar{\mathbf{Y}}\right)\bar{\mathbf{Y}}_{,t}\right)_\Omega.
\end{aligned} \tag{6.23}$$

Inserting (6.22) and (6.23) into (6.20) gives of our coarse-scale equation. We will restrict ourselves to one dimension (in anticipation of the forthcoming example) and write $\mathbf{A}_1$ as simply $\mathbf{A}$. Given $\mathbf{Y}'$, we seek $\bar{\mathbf{Y}} \in \bar{V}$ such that

$$a\left(\bar{\mathbf{W}};\bar{\mathbf{Y}}\right) - \left(\bar{\mathbf{W}}_{,x}, \mathbf{A}\left(\bar{\mathbf{Y}}\right)\mathbf{Y}'\right)_\Omega \mathbf{0} \quad \forall\bar{\mathbf{W}} \in \bar{V}. \tag{6.24}$$

For the fine-scale equation, we must work in physical entropy variables, $\mathbf{V}$, (see, *e.g.*, [50]). This is because the correct boundary conditions depend on the eigenvalues of matrices associated

with the entropy variables formulation, which is diagonalizable. Rather than identifying the inflow and outflow boundaries by the velocity alone as we have in our scalar cases, in the present context we must identify inward and outward pointing characteristics (see Hughes and Mallet [34]). Thus we have the following fine-scale problem where the coefficient matrices are evaluated at the coarse-scale solution, that is, $\mathbf{A} = \mathbf{A}\left(\bar{\mathbf{V}}\right)$:

$$-\left(\mathbf{W}'_{,x}, \tilde{\mathbf{A}}\mathbf{V}'\right)_{\Omega_e} + \frac{1}{2}\mathbf{W}' \cdot \left(\tilde{\mathbf{D}} + \tilde{\mathbf{M}}\right)\mathbf{V}'\Big|_{x_e}^{x_{e+1}} = \left(\mathbf{W}', \tilde{\mathbf{R}}'\left(\bar{\mathbf{Y}}\right)\right)_{\Omega_e}, \tag{6.25}$$

where

$$\tilde{\mathbf{A}} = \left.\frac{\partial \mathbf{F}^{\mathrm{adv}}}{\partial \mathbf{V}}\right|_{\mathbf{V}(\bar{\mathbf{Y}})} = \left.\left(\mathbf{A}\mathbf{Y}_{,\mathbf{V}}\right)\right|_{\bar{\mathbf{Y}}}, \tag{6.26}$$

$$\begin{aligned}\tilde{\mathbf{D}} &\equiv \begin{cases} (-1)\cdot\tilde{\mathbf{A}} & x = x_e \\ \tilde{\mathbf{A}} & x = x_{e+1} \end{cases} \\ &= \mathbf{S}\boldsymbol{\Lambda}\mathbf{S}^{-1}, \end{aligned} \tag{6.27}$$

$$\tilde{\mathbf{M}} \equiv \mathbf{S}\left|\boldsymbol{\Lambda}\right|\mathbf{S}^{-1}, \tag{6.28}$$

and $\boldsymbol{\Lambda}$ is the diagonal matrix of eigenvalues of $\tilde{\mathbf{D}}$, such that

$$\left[\mathbf{S}^{-1}\frac{1}{2}\left(\tilde{\mathbf{D}} + \tilde{\mathbf{M}}\right)\mathbf{S}\right]_{ij} = \max\left(\left[\boldsymbol{\Lambda}\right]_{ij}, 0\right). \tag{6.29}$$

This achieves an effect analogous to the χ^{out} term in the scalar examples – outward pointing characteristics pass, but inward pointing ones do not.

From this construction, we can obtain an element Green's function $\tilde{\mathbf{g}}_h^e$. To avoid the complexity of calculating and then integrating the residual in entropy variables, we calculate a stabilization matrix for entropy variables,

$$\tilde{\tau}_h = \frac{1}{\mathrm{meas}(\Omega_e)} \iint\limits_{\Omega_e \times \Omega_e} \tilde{\mathbf{g}}_h^e(x,y)\,dx\,dy. \tag{6.30}$$

This can be converted into an expression for primitive variables (see [29]) by the following transformation:

$$\tau_h = \mathbf{Y}_{,\mathbf{V}}\tilde{\tau}_h. \tag{6.31}$$

We now use a "PVMS-τ" method by letting $Y' = -\tau_h \mathbf{R}'(\bar{\mathbf{Y}})$.

The test case examined is that of a stationary shock in one dimension. We solve using the

above PVMS-τ approach and compare with the classical SUPG method. The mesh consists of 39 linear elements on $\Omega = (-19.5, 19.5)$. At the inflow, we choose $\rho_{in} = 1$, $u_{in} = 1$, and prescribe an inflow Mach number, M_{in}. The outflow conditions are given by the following relationships (see [30]):

$$\mathbf{M}_{out}^2 = \frac{\mathbf{M}_{in}^2 + \frac{2}{\gamma-1}}{\frac{2\gamma}{\gamma-1}\mathbf{M}_{in}^2 - 1}, \tag{6.32}$$

$$\frac{p_{out}}{p_{in}} = \frac{2\gamma}{\gamma+1}\mathbf{M}_{in}^2 - \frac{\gamma-1}{\gamma+1}, \tag{6.33}$$

$$\frac{\theta_{out}}{\theta_{in}} = \frac{\left(1 + \frac{\gamma-1}{2}\mathbf{M}_{in}^2\right)\left(\frac{2\gamma}{\gamma-1}\mathbf{M}_{in}^2 - 1\right)}{\frac{(\gamma+1)^2}{2(\gamma-1)}\mathbf{M}_{in}^2}. \tag{6.34}$$

The remaining variables are fixed by the constitutive equations and balance of the fluxes.

Figures 6.9 and 6.10 show pressure results for a Mach 2 shock and a Mach 3 shock, respectively. For the Mach 2 case, both SUPG and our PVMS-τ method offer reasonable solutions, with the PVMS-τ result appearing slightly diffusive. When the shock is strengthened to Mach 3, however, SUPG is not sufficiently stable. The oscillations become so large that density becomes negative, which immediately causes the Newton-Raphson iterations to fail to converge. The PVMS-τ approach has no such difficulty.

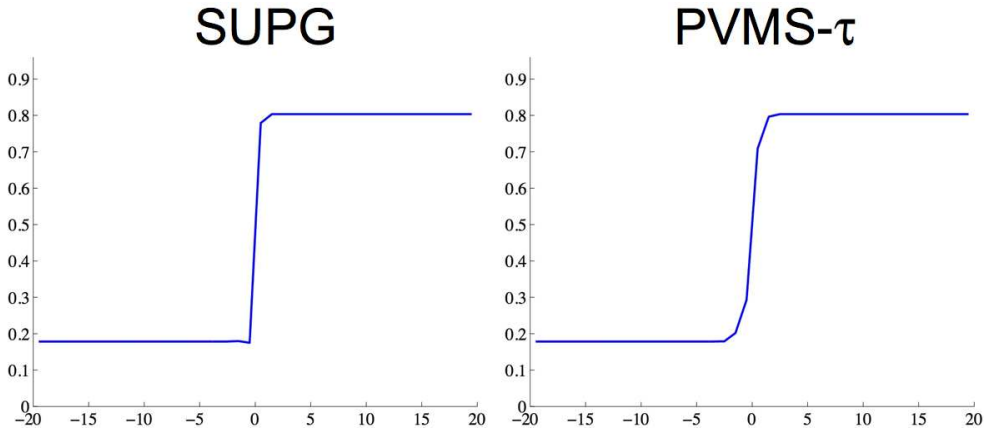


Figure 6.9: Mach 2 shock. Pressure results shown for SUPG and PVMS-τ.

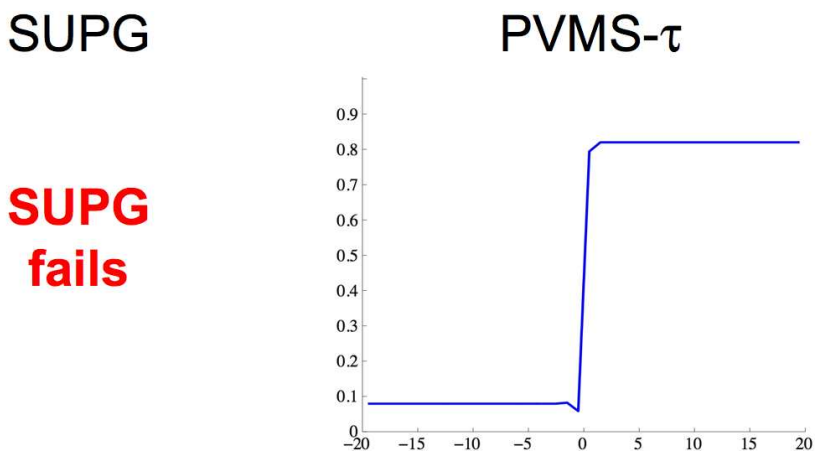


Figure 6.10: Mach 3 shock. SUPG fails to converge. PVMS experiences a very slight undershoot in the pressure to the left of the shock, but it is otherwise stable and has no difficulty converging.

Chapter 7

Conclusions

In this dissertation, the concept of isogeometric analysis has been presented. Its initial motivations were discussed, as were many details of our current NURBS based implementation, with a particular emphasis on refinement strategies. This methodology allows for the exact representation of a wide class of geometries on very coarse meshes, in particular conic sections can be represented exactly. Refinements can be performed by subdivision of the grid or by elevation of the polynomial order of the basis, all with an unprecedented level of control over the continuity of the basis functions and respect for the original geometry. Additionally, a method for local refinement has been presented. Numerous examples were shown that highlighted each of the major features of the current technology: geometric flexibility, functions of high continuity, and local refinement. These tests suggest that the method may be very well suited to a wide range of applications. Selected applications were presented to illustrate this point, and some current and future research directions were discussed.

The theoretical framework of the variational multiscale (VMS) method was presented as background for the current work in the field. Particular attention was given to the fine-scale Green's function and its role in defining the fine-scale solution. As an aside, the multiscale discontinuous Galerkin (MDG) method was discussed. Subsequently, these two topics were brought together as connections were made that allowed for the interpretation of MDG within the VMS framework. These observations served as the starting point for further attempts to numerically approximate the fine scales, specifically the fine-scale Green's function. A general framework for the numerical approximation of the fine-scale entities arising in VMS, including g', was presented – a first in the field. The advection-diffusion equation in both one and two dimensions served as our primary test case, allowing us to explore the efficacy of different approaches to modeling the fine scales. The effects of the choice of projections, polynomial order, patch size, and fine-mesh resolution were all examined.

The computational framework developed for exploring the fine scales provides opportunities

that were heretofore unavailable for the investigation of VMS, but at the expense of computational complexity. In an effort to provide a computationally feasible alternative, the parameter-free variational multiscale (PVMS) method was introduced. It simplifies and streamlines the machinery of the preceding sections to create a fast stabilization technique, born of a variational multiscale formulations, that does not require the *ad hoc* selection of stabilization parameters that has been the bane of much of stabilized methods research. The PVMS method was tested on linear and nonlinear problems for single equations as well as systems. Results indicate that it may be a very easily extensible approach to consistent, parameter-free stabilization of many types of equations.

Bibliography

[1] J. E. Akin and T. E. Tezduyar. Calculation of the advective limit of the SUPG stabilization parameter for linear and higher-order elements. *Computer Methods in Applied Mechanics and Engineering*, 193:1909–1922, 2004.

[2] I. Akkerman, Y. Bazilevs, V. Calo, T.J.R. Hughes, and S. Hulshoff. The role of continuity in residual-based variational multiscale modeling of turbulence. Technical report, ICES, The University of Texas at Austin, 2007.

[3] ARPACK. http://www.caam.rice.edu/software/arpack/.

[4] Y. Bazilevs, V. Calo, J.A. Cottrell, T.J.R. Hughes, A. Reali, and G. Scovazzi. Variational multiscale residual-based turbulence modeling for large eddy simulation of incompressible flows. Technical report, ICES, The University of Texas at Austin, 2007.

[5] Y. Bazilevs, V.M. Calo, Y. Zhang, and T.J.R. Hughes. Isogeometric fluid-structure interaction analysis with applications to arterial blood flow. *Computer Methods in Applied Mechanics and Engineering*, 38:310–322, 2006.

[6] Y. Bazilevs, L. Beirao de Veiga, J.A. Cottrell, T.J.R. Hughes, and G. Sangalli. Isogeometric analysis: approximation, stability and error estimates for h-refined meshes. *Mathematical Models and Methods in Applied Sciences*, 16:1031–1090, 2006.

[7] Y. Bazilevs and T.J.R. Hughes. Weak imposition of Dirichlet boundary conditions in fluid mechanics. *Computers and Fluids*, 36 (1):12–26, January 2007.

[8] J.K. Benninghof and R.B. Lehoucq. An automated multilevel substructuring method for eigenspace computations in linear elastodynamics. *SIAM Journal of Scientific Computing*, 25:2084–2106, 2004.

[9] M. Bischoff, W. A. Wall, K. -U. Bletzinger, and E. Ramm. Models and finite elements for thin-walled structures. In E. Stein, R. de Borst, and T. J. R. Hughes, editors, *Encyclopedia of*

Computational Mechanics, Vol. 2, Solids, Structures and Coupled Problems, chapter 3. Wiley, 2004.

[10] P. Bochev, T.J.R. Hughes, and G. Scovazzi. A multiscale discontinuous Galerkin method. *Lecture Notes in Computer Science*, 3743:84–93, 2006.

[11] F. Brezzi, B. Cockburn, L.D. Marini, and E. Suli. Stabilization mechanisms in discontinuous Galerkin finite element methods. *Computer Methods in Applied Mechanics and Engineering*, 195:3293–3310, 2006. In press.

[12] F. Brezzi, L. P. Franca, T. J. R. Hughes, and A. Russo. $b = \int g$. *Computer Methods in Applied Mechanics and Engineering*, 145:329–339, 1997.

[13] F. Brezzi and A. Russo. Choosing bubbles for advection-diffusion problems. *Mathematical Models and Methods in Applied Science*, 4:571–587, 1994.

[14] L. Brillouin. *Wave Propagation in Periodic Structures*. Dover Publications, Inc., Mineola, NY, 1953.

[15] A. N. Brooks and T. J. R. Hughes. Streamline upwind / Petrov-Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier-Stokes equations. *Computer Methods in Applied Mechanics and Engineering*, 32:199–259, 1982.

[16] R.D. Buehrle, G.A. Fleming, R.S. Pappa, and F.W. Grosveld. Finite element model development for aircraft fuselage structures. In proceedings of *XVIII International Modal Analysis Conference*, San Antonio, TX, 2000.

[17] A. Buffa, T.J.R. Hughes, and G. Sangalli. Analysis of a multiscal discontinuous Galerkin method for convection-diffusion problems. *SIAM Journal of Numerical Analysis*, 44 (4):1420–1440, 2006.

[18] J.A. Cottrell, T.J.R. Hughes, and A. Reali. Studies of refinement and continuity in isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering*, 2007. In press.

[19] J.A. Cottrell, A. Reali, Y. Bazilevs, and T.J.R. Hughes. Isogeometric analysis of structural vibrations. *Computer Methods in Applied Mechanics and Engineering*, 195:5257–5296, 2006.

[20] L. Couchman, S. Dey, and T. Barzow. *ATC Eigen-Analysis, STARS / ARPACK / NRL Solver*. Naval Research Laboratory, Englewood Cliffs, NJ, 2003.

[21] L. Demkowicz and J.T. Oden. An adaptive characteristic Petrov-Galerkin finite element method for convection-dominated linear and nonlinear parabolic problems in two space variables. *Computer Methods in Applied Mechanics and Engineering*, 55:63–87, 1986.

[22] G. Engel, K. Garikipati, T. J. R. Hughes, M. G. Larson, and L. Mazzei. Continuous /discontinuous finite element approximations of fourth-order elliptic problems in structural and continuum mechanics with applications to thin beams and plates, and strain gradient elasticity. *Computer Methods in Applied Mechanics and Engineering*, 191:3669–3750, 2002.

[23] C. Eskilsson and S. Sherwin. Spectral/hp discontinuous Galerkin methods for modelling 2D Boussinesq equations. *Journal of Computational Physics*, submitted.

[24] C. Farhat and B. Koobus. Finite volume discretization on unstructured meshes of the multiscale formulation of large eddy simulations. In F.G. Rammerstorfer, H.A. Mang, and J. Eberhardsteiner, editors, *In Proceedings of the Fifth World Congress on Computational Mechanics (WCCM V)*. Vienna University of Technology, Austria, July 7-12, 2002.

[25] G.E. Farin. *NURBS Curves and Surfaces: from Projective Geometry to Practical Use*. A. K. Peters, Ltd., Natick, MA, 1995.

[26] F.W. Grosveld, J.I. Pritchard, R.D. Buehrle, and R.S. Pappa. Finite element modeling of the NASA Langley Aluminum Testbed Cylinder. *8th AIAA/CEAS Aeroacoustics Conference*, Breckenridge, CO, 2002. AIAA 2002-2418.

[27] J.L. Guermond. A finite element technique for solving first-order pdes in l-p. *SIAM Journal of Numerical Analysis*, 42 (4):714–737, 2004.

[28] G. Hauke. Simple stabilizing matrices for the computation of compressible flows in primitive variables. *Computer Methods in Applied Mechanics and Engineering*, 190:6881–6893, 2001.

[29] G. Hauke and T. J. R. Hughes. A comparative study of different sets of variables for solving compressible and incompressible flows. *Computer Methods in Applied Mechanics and Engineering*, 153:1–44, 1998.

[30] P. Hill and C. Peterson. *Mechanics and Thermodynamics of Propulsion*. Addison Wesley, Reading, MA, 1992.

[31] T. J. R. Hughes. Multiscale phenomena: Green's functions, the Dirichlet-to-Neumann formulation, subgrid scale models, bubbles and the origins of stabilized methods. *Computer Methods in Applied Mechanics and Engineering*, 127:387–401, 1995.

[32] T. J. R. Hughes. *The Finite Element Method: Linear Static and Dynamic Finite Element Analysis*. Dover Publications, Mineola, NY, 2000.

[33] T. J. R. Hughes, G. Feijóo., L. Mazzei, and J. B. Quincy. The variational multiscale method – A paradigm for computational mechanics. *Computer Methods in Applied Mechanics and Engineering*, 166:3–24, 1998.

[34] T. J. R. Hughes and M. Mallet. A new finite element formulation for fluid dynamics: III. The generalized streamline operator for multidimensional advective-diffusive systems. *Computer Methods in Applied Mechanics and Engineering*, 58:305–328, 1986.

[35] T. J. R. Hughes, L. Mazzei, and K. E. Jansen. Large-eddy simulation and the variational multiscale method. *Computing and Visualization in Science*, 3:47–59, 2000.

[36] T. J. R. Hughes, G. Scovazzi, and L. P. Franca. Multiscale and stabilized methods. In E. Stein, R. De Borst, and T. J. R. Hughes, editors, *Encyclopedia of Computational Mechanics, Vol. 3, Computational Fluid Dynamics*, chapter 2. Wiley, 2004.

[37] T. J. R. Hughes and J. Stewart. A space-time formulation for multiscale phenomena. *J. Comput. Appl. Math.*, 74:217–229, 1996.

[38] T.J.R. Hughes, J.A. Cottrell, and Y. Bazilevs. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry, and mesh refinement. *Computer Methods in Applied Mechanics and Engineering*, 194:4135–4195, 2005.

[39] T.J.R. Hughes and G. Sangalli. Variational multiscale analysis: the fine-scale Green's function, projection, optimization, localization, and stabilized methods. Technical report, ICES, The University of Texas at Austin, 2005.

[40] T.J.R. Hughes, G. Scovazzi, P. Bochev, and A. Buffa. A multiscale discontinuous Galerkin method with the computational structure of a continuous Galerkin method. *Computer Methods in Applied Mechanics and Engineering*, 195:2761–2787, 2006.

[41] K.E. Jansen and A.E. Tejada-Martinez. An evaluation of the variational multiscale model for large-eddy simulation while using a hierarchical basis. In *AIAA Paper 2002-0283*, 2002.

[42] P. Kagan, A. Fischer, and P. Z. Bar-Yoseph. New B-spline finite element approach for geometrical design and mechanical analysis. *International Journal of Numerical Methods in Engineering*, 41:435–458, 1998.

[43] P. Kagan, A. Fischer, and P. Z. Bar-Yoseph. Mechanically based models: Adaptive refinement for B-spline finite element. *International Journal of Numerical Methods in Engineering*, 57:1145–1175, 2003.

[44] A. Masud and R. A. Khurram. A multiscale/stabilized finite element method for the advection-diffusion equation. *Computer Methods in Applied Mechanics and Engineering*, 193:1997–2018, 2004.

[45] D. Natekar, X. F. Zhang, and G. Subbarayan. Constructive solid analysis: a hierarchical, geometry-based meshless analysis procedure for integrated design and analysis. *Computer-Aided Design*, 36(5):473–486, 2004.

[46] L. Piegl and W. Tiller. *The NURBS Book (Monographs in Visual Communication), 2nd ed.* Springer-Verlag, New York, 1997.

[47] E. Rank, A. Düster, V. Nübel, K. Preusch, and O. T. Bruhns. High order finite elements for shells. *Computer Methods in Applied Mechanics and Engineering*, 194 (21-24):2494–2512, 2005.

[48] D. F. Rogers. *An Introduction to NURBS With Historical Perspective*. Academic Press, San Diego, CA, 2001.

[49] T.N. Sederberg, J.M. Zhengs, A. Bakenov, and A. Nasri. T-splines and T-NURCCSs. *ACM Transactions on Graphics*, 22 (3):477–484, 2003.

[50] F. Shakib, T. J. R. Hughes, and Z. Johan. A new finite element formulation for computational fluid dynamics: X. The compressible Euler and Navier-Stokes equations. *Computer Methods in Applied Mechanics and Engineering*, 89:141–219, 1991.

[51] D.C. Simkins, A. Dumar, N. Collier, and L.B. Whitenack. Geometry representation, modification and iterative design using RKEM. *Computer Methods in Applied Mechanics and Engineering*, Submitted.

[52] G. Strang and G.J. Fix. *An Analysis of the Finite Element Method*. Prentice-Hall, Englewood Cliffs, NJ, 1973.

[53] K. S. Surana, A. R. Ahmadi, and J. N. Reddy. The *k*-version finite element method for self-adjoint operators in BVP. *International Journal of Computational Engineering Science*, 3:155–218, 2002.

[54] K. S. Surana, R. K. Maduri, P. W. TenPas, and J. N. Reddy. Elastic wave propagation in laminated composites using the space-time least-squares formulation in *h*, *p*, *k* framework. *Mechanics of Advanced Materials and Structures*, 13:161–196, 2006.

[55] B. Szabó, A. Düster, and E. Rank. The p-version of the finite element method. In E. Stein, R. de Borst, and T. J. R. Hughes, editors, *Encyclopedia of Computational Mechanics, Vol. 1, Fundamentals*, chapter 5. Wiley, 2004.

[56] S. Timoshenko and S. Woinowsky-Krieger. *Theory of Plates and Shells (Engineering Societies Monograph), 2nd ed.* McGraw-Hill, 1959.

[57] L. B. Wahlbin. Local behavior in finite element methods. In P. G. Ciarlet and J. L. Lions, editors, *Finite Element Methods (Part 1)*, volume 2 of *Handbook of Numerical Analysis*, pages 353–522. North-Holland, 1991.

[58] X. F. Zhang and G. Subbarayan. jNURBS: An object-oriented, symbolic framework for integrated, meshless analysis and optimal design. *Advances in Engineering Software*, 37(5):287–311, 2006.

[59] Y. Zhang, Y. Bazilevs, S. Goswami, C. Bajaj, and T.J.R. Hughes. Patient-specific vascular NURBS modeling for isogeometric analysis of blood flow. In 15^{th} *International Meshing Roundtable Conference*, pages 73–92, Birmingham, AL, September 17-20, 2006. Accepted for publication in *Computer Methods in Applied Mechanics and Engineering*.

Vita

John Austin Cottrell, III, son of Dr. and Mrs. John A. Cottrell, Jr., was born on August 1st in the year 1980, in Woodstock, Virginia. After graduating as the Valedictorian of Central High School in 1998, he enrolled at Rice University in Houston, TX. In May 2002, he received a Bachelor of Science in Physics. He subsequently moved to Austin to pursue a PhD degree in Computational and Applied Mathematics (CAM) at the Institute for Computational Engineering and Sciences, where he received his Master of Science degree in 2004. He has been in school without interruption since 1985 and hopes that the light at the end of the tunnel is not a train.

Permanent Address: 989 Black Bear Rd.
Maurertown, Virginia 22644

This dissertation was typeset with LaTeX 2_ε[1] by the author.

[1] LaTeX 2_ε is an extension of LaTeX. LaTeX is a collection of macros for TeX. TeX is a trademark of the American Mathematical Society. The macros used in formatting this dissertation were written by Dinesh Das, Department of Computer Sciences, The University of Texas at Austin, and extended by Bert Kay, James A. Bednar, and Ayman El-Khashab.